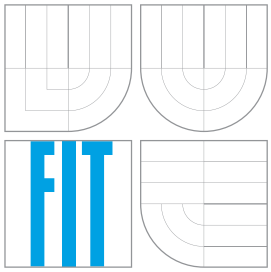
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SYSTÉM PRO SPRÁVU A KOMUNIKACI HLÁŠENÍ O PROBLÉMECH

TROUBLE TICKET MANAGEMENT AND COMMUNICATION SYSTEM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR
TOMÁŠ POREMBA

VEDOUCÍ PRÁCE
SUPERVISOR
RNDr. MAREK RYCHLÝ, Ph.D.

BRNO 2016

Zadání bakalářské práce/18907/2015/xporem00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2015/2016

Zadání bakalářské práce

Řešitel: **Poremba Tomáš**
Obor: Informační technologie
Téma: **Systém pro správu a komunikaci hlášení o problémech**
Trouble Ticket Management and Communication System
Kategorie: Databáze

Pokyny:

1. Seznamte se s problematikou správy incidentů a problémů a s existujícími systémy pro správu hlášení o incidentech či problémech. Vybrané systémy porovnejte.
2. Navrhněte vlastní systém pro správu a komunikaci hlášení o problémech. Systém bude spravovat životní cyklus hlášení, komunikovat s uživateli přes webové rozhraní a email, s externím systémem pro správu projektů (např. Citrix Podio) a generovat přehledy postupu a výkonnosti zpracování hlášení.
3. Po konzultaci s vedoucím systém implementujte s využitím rámce AngularJS.
4. Výslednou implementaci důkladně otestujte a pokud možno nasaďte do zkušebního provozu.
5. Proveďte zhodnocení dosažených výsledků a diskutujte další možný vývoj projektu.

Literatura:

- David Cannon, David Wheeldon. *ITIL Service Operation*. The Stationery Office, 2007. ISBN 978-0-11-331046-3.
- Pawel Kozlowski, Peter Bacon Darwin. *Mastering Web Application Development with AngularJS*. Packt Publishing, 2013. ISBN 978-1782161820.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 a 2, a započatá práce na řešení bodu 3.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Rychlý Marek, RNDr., Ph.D.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2015
Datum odevzdání: 18. května 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Dušan Kolář
vedoucí ústavu

Abstrakt

Táto bakalárska práca sa zaoberá návrhom a implementáciou systému pre správu a komunikáciu hlásení o problémoch. Je v nej rozobratá problematika správy incidentov a problémov a dokumentácia ich životných cyklov. Práca porovnáva existujúce riešenia zaoberajúce sa danou problematikou. Vysvetlenie problematiky sa zameriava na podprocesy nutné pre úspešné zvládnutie správy hlásení. Procesy sú zahrnuté v návrhu a implementácii systému, ktorý využíva platformu .Net a JavaScriptový aplikačný rámec AngularJS. Časť implementácie používa integráciu so systémom na správu projektov Podio. Systém je vyvíjaný v spolupráci so spoločnosťou WebStudy a jeho vývoj rešpektuje zaužívané postupy v tejto spoločnosti.

Abstract

This bachelor thesis deals with a design and an implementation of a ticket management and coummunication system. It explains the problem of incident and problem management and a documentation of their lifecycles. The thesis compares existing solutions that handle the problem. A description of the problem focuses on subprocesses needed for successful mastering of incident and problem management. These processes are included in the design and the implementation of a system. The system is built with a help of .Net platform and JavaScript framework AngularJS. Part of the implementation uses integration with a project management system Podio. The system is developed in co-operation with the company WebStudy and its development respects exerted policies.

Kľúčové slová

informačný systém, incident, problém, správa, užívateľská podpora

Keywords

information system, incident, problem, management, technical support

Citácia

POREMBA, Tomáš. *Systém pro správu a komunikaci hlášení o problémech.* Brno, 2016. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Rychlý Marek.

Systém pro správu a komunikaci hlášení o problémech

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Marka Rychlého. Ďalšie informácie mi poskytla spoločnosť WebStudy EU s.r.o. a jej vedúci, pán Pavel Šanca. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....
Tomáš Poremba
24. mája 2016

Poďakovanie

Na tomto mieste by som rád poďakoval vedúcemu mojej práce, pánovi Markovi Rychlému za odborné rady a pomoc pri riešení problémov spojených s vypracovaním tejto práce. Poďakovanie patrí tiež spoločnosti WebStudy EU s.r.o., v spolupráci s ktorou mohlo toto dielo vzniknúť a hlavne vedúcemu tejto spoločnosti, pánovi Pavlovi Šancovi.

Obsah

Kapitola 1

Úvod

Vývoj a návrh softvéru je iba malou časťou jeho životného cyklu alebo služby, ktorú poskytuje. Prevádzkovanie služby zaberá dlhší časový úsek a jej udržovanie v stave, v ktorom ju užívatelia môžu používať, je rovnako potrebné ako samotný vývoj. V dnešnej dobe môže nedostupnosť poskytovanej služby zákazníka stáť nemalé finančné prostriedky, a preto je promptné odhaľovanie a riešenie príčin nedostupnosti nevyhnutné. Spoločnosti, ktoré dané služby prevádzkujú, sa tak zameriavajú aj na bezchybnú prevádzku služieb a odstraňovanie prekážok v dostupnosti daných služieb. Nemalou časťou tohoto procesu je manažment incidentov a problémov v oblasti užívateľskej podpory. Ten je prvým procesom, s ktorým sa užívateľ stretáva pri riešení prekážok v dostupnosti služieb a tvorí nástroj, pomocou ktorého sa služby navracajú k normálnej prevádzke.

Cieľom tejto práce je navrhnúť a implementovať systém pre správu incidentov a problémov v systéme pre správu vzdelávania (LMS) spoločnosti WebStudy s ohľadom na zaužívané postupy v spoločnosti. Systém by mal zjednodušiť a sprehľadniť komunikáciu ohľadom nedostupnosti alebo zhoršenia kvality poskytovaných služieb a prispieť k urýchleniu návratu služieb do normálnej prevádzky. Súčasťou práce je vysvetlenie problematiky správy incidentov a problémov, analýza existujúcich riešení a ich zhodnotenie pre potreby spoločnosti, návrh nového systému a popis implementačných detailov v častiach predstavujúcich základné časti systému. Návrh a implementácia nového riešenia musí zohľadňovať postupný vývoj aplikácie, umožniť jednoduchú integráciu novovyvinutých častí a takisto poskytovať možnosti na využitie už prevádzkovaných častí aplikácie.

Spoločnosť WebStudy vznikla v roku 1999. Od svojho počiatku poskytuje služby v oblasti e-learningu. Jej hlavným produktom je rovnomenný systém na riadenie výuky poskytovaný ako cloudová služba – webová aplikácia. WebStudy si zakladá na úzkej spolupráci so svojimi zákazníkmi a ich spokojnosti, a aj preto sa rozhodla inovovať proces, ktorý zlepšuje kvalitu poskytovanej služby.

Kapitola 2

Úvod do problematiky

Úlohou tejto práce je návrh a implementácia systému pre správu incidentov a problémov. Systém má poskytnúť nástroj uľahčujúci procesy prevádzkovania služieb poskytovaných spoločnosťou WebStudy.

Pod pojmom **služba** budeme rozumieť prostriedok dodania hodnoty zákazníkovi, ktorá umožňuje zákazníkovi dosiahnuť také výsledky, aké potrebuje. Podstatnou časťou služby je, že neprenáša na zákazníka náklady a riziká spojené s prevádzkovaním služby. **Služba IT** je špeciálny typ služby, ktorá využíva informačné technológie [9]. V tejto práci budeme s ohľadom na zadanie pojem služba IT pre zjednodušenie nazývať služba.

Vzhľadom na charakter poskytovanej služby, pre ktorú bude systém vytvorený, budeme daný systém nazývať informačný. Pre správne riešenie problému je nevyhnutné pochopenie jednotlivých pojmov. Tie budú vysvetlené na nasledujúcich riadkoch.

2.1 Informačný systém

Pojem informačný systém sa skladá z dvoch častí. Aby sme pochopili význam tohto slovného spojenia, potrebujeme si vysvetliť význam slov, z ktorých sa skladá.

Vo všeobecnom význame za **informáciu** považujeme údaj o reálnom prostredí, o jeho stave a procesoch v ňom prebiehajúcich. Z pohľadu výpočtovej techniky môžeme informáciu chápať ako kvantitatívne vyjadrenie obsahu správy [2]. Pre naše potreby môžeme zjednodušene povedať, že informácia je interpretovaný údaj alebo interpretované dáta.

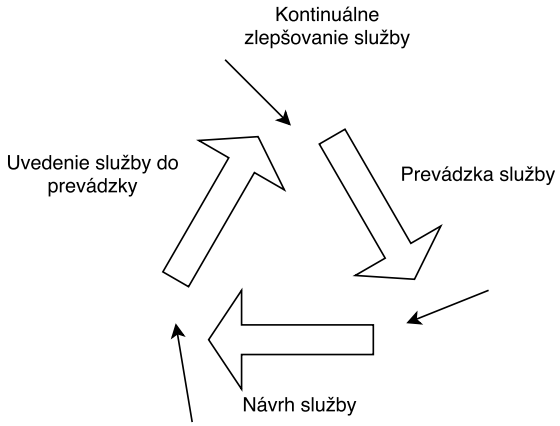
Podľa [5] je **systém** množina prvkov a väzieb medzi nimi, definovaná na nejakom nosiči. Nosič je množina prvkov vo vzájomných informačných a procesných vzťahoch. Prvky nosiča nazývame zdroje. Neoddeliteľnou charakteristikou systému je jeho stav. Stavom systému rozumieme hodnoty jeho zdrojov. Zdroje môžeme rozdeliť na:

- fyzické,
- konceptuálne (pojmové) – informácie.

Práve konceptuálne zdroje nás budú s ohľadom na pojem informačný systém zaujímať. **Informačný systém** je systém modelujúci fyzický systém a využívajúci konceptuálne zdroje, teda informácie vo forme dát [5].

2.2 Životný cyklus služby a podpora užívateľov

Aby bola služba pre užívateľov užitočná, je potrebné každej etape životného cyklu služby venovať dostatočnú pozornosť. Zo všetkých etáp má totiž prevádzkovateľ služby možnosť získať informácie, ktoré pomôžu službu vylepšiť alebo zefektívniť. Nemusí to byť len vhodný, efektívny a udržateľný návrh vytvorený po definovaní zákazníckych požiadaviek, ale môžu to byť aj poznatky z nasadzovania služby do prevádzky a hlavne spätná väzba od užívateľov danej služby. Práve zber a vyhodnotenie spätnej väzby v rôznych podobách je časť z úloh, ktoré náležia procesom prevádzkovania služby. Medzi tieto patrí správa udalostí, správa incidentov a problémov, spĺňanie požiadaviek a správa prístupu. Pre účely tejto bakalárskej práce sa budeme zaoberať správou incidentov a problémov a rozšírime procesy aj o zber užívateľskej spätnej väzby vo forme užívateľských požiadaviek. Pre naše potreby nebudú mať užívateľské požiadavky význam v etape prevádzkovania služby, budú to skôr nápady a žiadosti o zlepšenie vyvíjanej aplikácie (viď 2.4), tj. budú spadať pod etapu uvedenia služby do prevádzky.



Obr. 2.1: Životný cyklus služby[1]

Môžeme teda povedať, že úlohou prevádzky služby je doručenie služby efektívne a účinne. To zahŕňa udržanie služby v normálnom chode.Normálnym chodom budeme rozumieť stav služby, ktorý spĺňa podmienky stanovené v SLA (z angl. service-level agreement – dohoda medzi poskytovateľom a užívateľom služby o rozsahu, cene a kvalite poskytovanej služby) [8]. Kvalitu služby môžu ovplyvňovať rôzne udalosti, ktorými sa zaoberá proces správy udalostí. Ak majú užívatelia pocit, že kvalita služby nevyhovuje dohodnutej úrovni, mali by mať možnosť obrátiť sa so svojimi pripomienkami na určitú entitu, ktorá sa nimi bude zaoberať. Touto entitou je tzv. service desk. Hlavnou úlohou service desku je tak vytvoriť kontaktné miesto na komunikáciu pre zákazníka aj zamestnanca spoločnosti. Predstavuje teda formu **užívateľskej podpory**.

Udalosti, ktoré negatívne ovplyvňujú kvalitu služby sa nazývajú incidenty. Tým sa budem venovať v ďalšej podkapitole, ktorej základným zdrojom informácii je [8].

[1] Upravené podľa [8].

2.3 Incidenty, problémy a ich správa

Podľa odporúčaní ITIL v prevádzke služby funguje správa udalostí. **Udalosťou** budeme rozumieť každú zmenu stavu systému, ktorá má na neho signifikantný dopad. Každá udalosť, ktorá ovplyvní systém natoľko, že kvalita poskytovanej služby klesne pod úroveň dohodnutú v SLA, sa stáva **incidentom**. Jednou z úloh správy udalostí je predchádzať takým udalostiam, ktoré by sa transformovali až na incidenty, no je prakticky nemožné všetkým incidentom predísť. Tým pádom pri poskytovaní služby musí existovať proces, ktorý sa danou skupinou udalostí bude zaoberať. Týmto procesom je správa incidentov.

2.3.1 Správa incidentov

Každý poskytovateľ služby by sa incidentom rád vyhol, no je to prakticky nereálne. Musí preto zabezpečiť mechanizmus, ktorý sa postará o evidenciu a spracovanie všetkých vzniknutých incidentov. Pod pojmom **správa incidentov** rozumieme proces, ktorý sa zaoberá riadením životného cyklu všetkých incidentov. Incidenty môže rozpoznať buď technický personál spoločnosti poskytujúcej službu, systém na správu udalostí alebo aj koncový užívateľ.

Význam a ciele

Hlavným cieľom správy incidentov je navrátenie služby do normálneho chodu čo najrýchlejšie a obmedzenie dopadu vzniknutého incidentu na zákazníkov biznis. Zaručuje dodržanie dohodnutej úrovne SLA. Ďalej poskytuje nástroje na odkomunikovanie vzniknutých incidentov s technickým i obchodným personálom. Musí takisto správne identifikovať incidenty a odlíšiť ich od ostatných druhov užívateľských vstupov, napr. požiadaviek na službu.

Kľúčové požiadavky

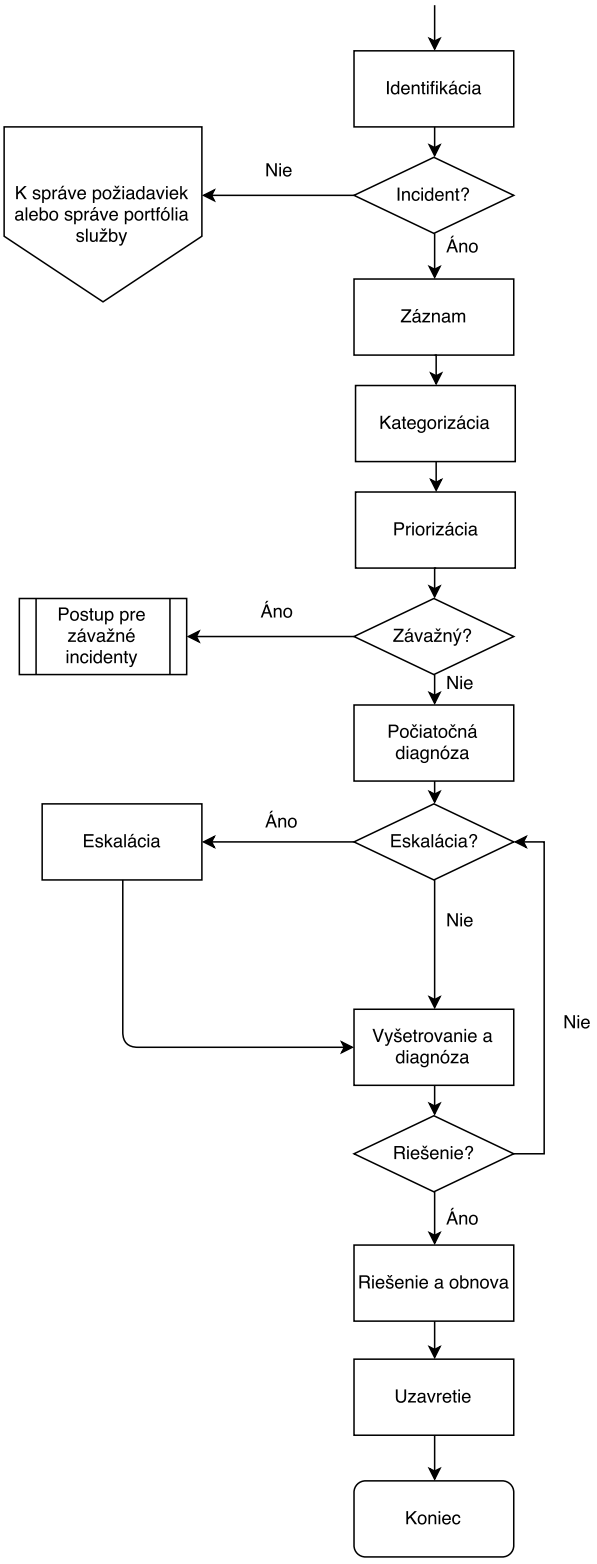
Pre správnu prevádzku správy incidentov sú nevyhnutné nasledujúce skutočnosti:

- **včasná komunikácia o incidentoch a ich stave**,
- včasné vyriešenie incidentu,
- udržanie zákazníckej spokojnosti,
- periodický audit incidentov, ich kategórií, priorít alebo eskalácie,
- **uloženie všetkých záznamov incidentov v jednom systéme**,
- **jednotný formát všetkých záznamov**.

Dodržanie zvýraznených položiek zoznamu nie je len úlohou členov service desku, ale pri ich splnení pomáha aj implementovaný systém. Už samotný návrh systému teda musí tieto body vziať do úvahy.

Podprocesy zahrnuté v správe incidentov

Správa incidentu je rozdelená na podprocesy, ktorých úlohou je udržať jednotný postup pri zaznamenaní a neskoršom riešení incidentu. Dodržanie týchto podprocesov zrýchľuje dosiahnutie požadovaného výsledku, ktorým je prinavrátenie kvality služby užívateľovi na



Obr. 2.2: Diagram správy incidentov v spoločnosti WebStudy [8]

požadovanú úroveň. Znázornenie náväznosti jednotlivých podprocesov sa nachádza na obrázku 2.2.

Identifikácia
Ideálnou situáciou, je narazenie na a vyriešenie incidentu ešte predtým, ako incident ovplyvní užívateľa. Ak však užívateľ postrehne ovplyvnenie služby, hlási to poskytovateľovi služby. Úlohou podporného tímu je rozlíšenie skutočného typu užívateľovej pripomienky – nemusí sa totiž nutne jednať o incident. Ak však užívateľova pripomienka incidentom je, pokračuje sa nasledujúcim procesom.

Záznam incidentu
Všetky incidenty musia byť zadokumentované. Nezáleží na tom, či daný incident nahlásil užívateľ, systém pre správu udalostí alebo personál spoločnosti, záznam o incidente musí existovať. Obsahovať by mal relevantné informácie vzhľadom na svoju podstatu, aby s ním mohli pracovať aj ostatní členovia podporného tímu, nie len tí, ktorí ho zaregistrovali a zaznamenali do systému. Medzi relevantné informácie patrí:

- unikátne identifikačné číslo,
- kategória,
- priorita,
- dátum a čas vytvorenia záznamu,
- identifikácia osoby, vytvárajúcej záznam,
- kontakt na osobu, vytvárajúcu záznam,
- popis incidentu,
- člen podporného tímu zaoberajúci sa incidentom,
- podstúpené kroky k vyriešeniu incidentu,
- aktuálny stav postupu riešenia.

Kategorizácia
Dôležitou súčasťou vytvorenia záznamu je priradenie správnej kategórie incidentu. Efektívna kategorizácia pomôže pri neskoršom zbere informácií o najproblematickejších častiach služby a vyhodnocovaní úspešnosti správy incidentov. Taktiež môže pomôcť pri priradení člena podporného tímu, ktorý má s riešením danej kategórie incidentov najväčšie skúsenosti.

Priorizácia
Záznam o incidente musí ďalej obsahovať jeho prioritu. Tá určuje, do akej miery je užívateľ poškodený vzniknutým incidentom a mala by vymedziť približný čas, za ktorý je potrebné nájsť riešenie incidentu. Priorita môže byť dynamická, napr. ak sa nepodarí daný incident vyriešiť načas, môže význam incidentu poklesnúť. Takisto poskytuje mechanizmus na uprednostnenie požiadaviek určitej, významnejšej, skupiny užívateľov.

Prvotná diagnostika
Ak bol vzniknutý incident zaznamenaný telefónnou linkou, a teda je člen podporného tímu

stále v kontakte s užívateľom, mal by o danom incidente zistiť od užívateľa čo najviac informácií – vytýčiť symptómy, na ktoré užívateľ narazil a čo najpresnejšie určiť miesto, kde nastala chyba. Pokiaľ je to v jeho možnostiach a riešenie incidentu nevyžaduje čas presahujúci dĺžku telefónneho hovoru, mal by incident vyriešiť hneď a po dohode a súhlase užívateľa daný incident aj uzavrieť.

Eskalácia
Každý incident sa však nedá vyriešiť okamžite. Akonáhle sa zistí, že nájdenie riešenia alebo jeho aplikácia nie je v kompetencii člena podporného tímu, musí byť incident postúpený tímu alebo inému členovi disponujúcemu prostriedkami na vyriešenie incidentu. Eskalácia incidentu neznamená, že sa zmení majiteľ záznamu incidentu. Záznam ako taký stále patrí členovi podporného tímu a ten je zodpovedný za jeho riešenie, komunikáciu s užívateľom a uzavretie incidentu.

Vyšetrovanie incidentu a jeho diagnostika
Hľadanie riešenia incidentu sa nezaobíde bez určitej miery skúmania skutočnej podstaty incidentu. Priradený člen podporného tímu tak musí zistiť čo najviac relevantných informácií, ktoré napomôžu jemu alebo inému delegovanému pracovníkovi vyriešiť incident. Všetky kroky a zistenia sú potom poznačené v zázname o incidente. Medzi vyžadované akcie patrí určenie:

- skutočnej podstaty incidentu,
- užívateľových predošlých aktivít, ktoré mohli mať efekt na vznik incidentu,
- rozsiahlosti incidentu a jeho dopadu na užívateľov,
- podrobný výskum problémov a databázy známych chýb.

Vyriešenie incidentu a obnovenie služby
Po nájdení potenciálneho riešenia ho člen podporného tímu použije a dôkladne otestuje. Keďže podstata incidentu nemusí zahŕňať iba prostriedky a zdroje pod priamou kontrolou spoločnosti prevádzkujúcej službu, kroky, ktoré musia byť pri riešení incidentu podstúpené môže vykonať aj člen podporného tímu na strane užívateľa (buď žiadosťou o ich vykonanie, alebo vzdialeným prístupom po dohode s užívateľom) alebo kontaktovaním poskytovateľa externej služby, ktorá daný incident vyvolala. Dôležitou súčasťou riešenia problému a návratu služby do normálneho chodu je dôkladné testovanie poskytnutého riešenia. Pomocou neho sa môžu obe zainteresované strany aspoň do istej miery uistiť, že daný incident bol skutočne odstránený poskytnutým riešením. Ak boli tieto kroky uskutočnené delegovaným tímom alebo členom podporného tímu, tento vráti incident pôvodnému majiteľovi, ktorý pristúpi k uzavretiu incidentu.

Uzavretie incidentu
Po tom, čo sa poskytnuté riešenie prejaví ako správne a užívateľ s ním súhlasí, môže byť incident uzavrený. Člen podporného tímu potom môže overiť správnu kategorizáciu incidentu, zistiť nakoľko bol užívateľ spokojný s priebehom riešenia a komunikáciou, prípadne zdokumentuje významné kroky vedúce k nájdeniu riešenia. Nakoniec je incident formálne uzavrený.

Ukazovatele úspešného zvládnutia správy incidentov

Správa incidentov môže spoločnosti poskytnúť dôležité informácie o kvalite poskytovanej služby, jej nedostatkoch alebo o výkone jej zamestnancov. Tie sa pre každú službu líšia a sú špecifické pre každú spoločnosť. V spoločnosti WebStudy sú podstatné tieto ukazovatele:

- rýchlosť úspešného vyriešenia incidentov,
- udržanie kvality služby – počet incidentov, počet závažných incidentov,
- správna kategorizácia incidentov – počet incidentov správne identifikovaných pri ich vzniku (správna kategorizácia poslúži aj na identifikáciu problematických miest služby).

Zhrnutie

Úspešné zvládnutie správy incidentov je kľúčové pre udržanie kvality poskytovanej služby. Vyžaduje však plné nasadenie členov podporného tímu, presvedčenie, že každý incident potrebuje svoju pozornosť a udržovanie prehľadu známych problémov a chýb. Tieto faktory významne pomáhajú pri zrýchlení procesu riešenia incidentu a pri zvýšení spokojnosti koncového užívateľa. Podstatným prvkom v zlepšovaní správy je aj efektívny systém, ktorý umožní pohodlnú prácu členom podporného tímu a zabezpečí plynulú komunikáciu s užívateľom.

Pre potreby spoločnosti WebStudy je správa incidentov dôležitou súčasťou procesu prevádzkovania služby. Navrhovaný systém bude teda čo možno najpodrobnejšie implementovať správu incidentu a jeho životného cyklu v rozsahu popísanom na predchádzajúcich riadkoch.

2.3.2 Správa problémov

V tejto podkapitole bude vysvetlená problematika správy problémov v kontexte požiadaviek a zaužívaných postupov v spoločnosti WebStudy.

Pod pojmom problém rozumieme príčinu jedného alebo viacerých incidentov. Správa problémov je zodpovedná za riadenie životného cyklu problému. Jej úlohou je proaktívne i reaktívne riešiť odhalené problémy, dokumentovať ich a popisovať známe chyby.

Proaktívny prístup k správe problémov sa odlišuje od prístupu správy incidentov. Charakter incidentov spôsobuje, že o nich pred nahlásením nevieme. Problémy sa však dajú odhaliť v rámci aktivít hľadajúcich cesty k zlepšeniu služby.

Význam a ciele

Cieľom správy problémov je predchádzanie vzniku nových problémov a na nich nadväzujúcich incidentov, minimalizácia počtu opakujúcich sa incidentov a zmiernenie dopadu incidentov na kvalitu služby. Dobrá správa problémov zvýši dostupnosť služby a jej poskytovanú kvalitu zákazníkom. Tiež uľahčí prácu členom podporného tímu vďaka podrobnej dokumentácii problémov. Tá urýchli riešenie príbuzných incidentov a zmenší počet nových náhradných riešení incidentov.

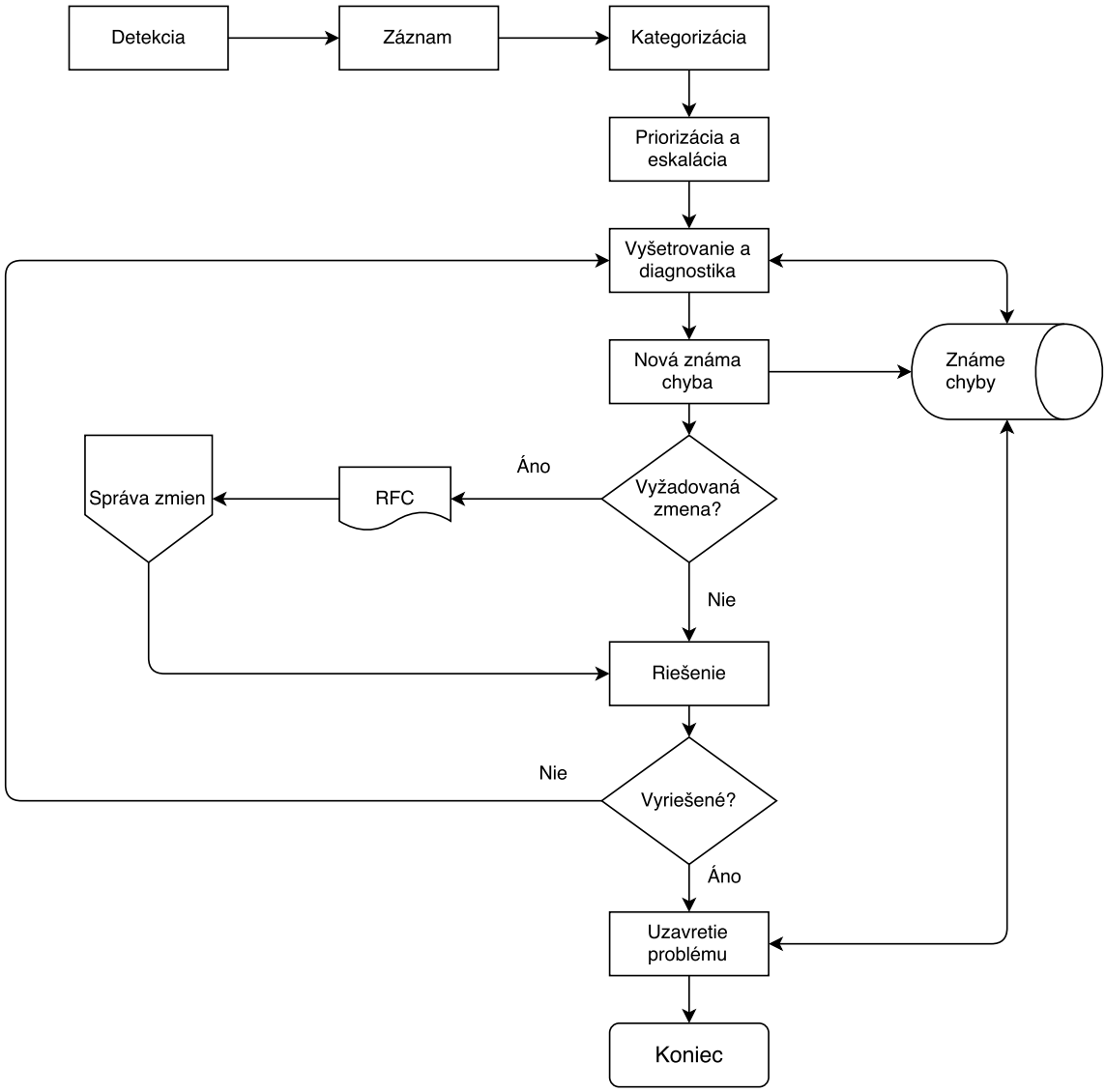
Kľúčové požiadavky

Kľúčové požiadavky na správu problémov sa v podstate nelíšia od požiadaviek správy incidentov. Podľa ITIL je naviac vhodné, aby sa záznamy o problémoch neplietli so záznamami

incidentov. Túto požiadavku však spoločnosť WebStudy nespĺňa a prehľad problémov sa snaží udržovať v blízkosti prehľadu incidentov, hlavne pre ich puto a jednoduché prepájanie incidentov a problémov.

Podprocesy zahrnuté v správe problémov

Správa problémov, podobne ako správa incidentov, sa delí na jednotlivé podprocesy, ktoré majú za úlohu životný cyklus problému urýchliť a sprehľadniť. Postupnosť týchto procesov je znázornená na obrázku 2.3.



Obr. 2.3: Diagram správy problémov upravený pre potreby spoločnosti WebStudy [8]

Detekcia problému

Problém môže byť detekovaný viacerými cestami. Najčastejšou je reakcia na vzniknutý incident. Ak si členovia podporného tímu nie sú istí jasnou príčinou vzniku incidentu, je potrebné nájsť problém, ktorý ho spôsobil. Problémy sa dajú detekovať aj proaktívnym

spôsobom. Skúmaním vzniknutých incidentov alebo sledovaním trendov incidentov v súvislosti so zmenami v aplikácii sa dajú objaviť zdroje problémov.

Záznam problému
Každý problém musí byť v systéme poznamenaný. Podstatné informácie zahŕňajú detaily incidentu, ktorý daný problém odhalil, kategóriu a prioritu problému, ďalšie súvisiace incidenty a pod.

Kategorizácia
Kategorizácia problému uľahčuje neskoršie vyhľadanie daného problému, pripojenie súvisiacich incidentov a vystihnutie podstaty problému.

Priorizácia a eskalácia
Priorizácia problému poukazuje na význam jeho dopadu na chod systému. Poradie riešenia problémov sa potom riadi závažnosťou jednotlivých problémov. Tá sa odvíja od viacerých faktorov – počet postihnutých užívateľov, význam zákazníka a pod. V tomto kroku spoločnosť WebStudy pridáva ako podproces aj eskaláciu. Vzhľadom na prepojenosť jednotlivých tímov podpory a vývoja sa často stáva, že problémy sú schopné vyriešiť rôzne tímy. Mechanizmus eskalácie problému priradí riešenie problému tímu, ktorý je dostatočne, či už časovo alebo schopnosťami, kompetentný.

Vyšetrovanie problému a diagnostika
Po predchádzajúcich krokoch nasleduje riešenie problému s najvyššou prioritou. To vyžaduje preskúmanie spojených incidentov, overenie funkčnosti podozrivých častí aplikácie a znovu vytváranie incidentov. Podrobný výskum napomôže nájdeniu najlepšieho riešenia daného problému. Ak sa odhalí skutočná podstata problému, musí byť dobre zdokumentovaná. Na to už spoločnosť WebStudy má vytvorené zdroje a vytváranie databázy známych chýb teda nebude predmetom vytváraného systému. Akonáhle sa zistí, že odstránenie problému vyžaduje zmenu systému, je návrh na zmenu postúpený správe zmien.

Riešenie problému
Na základe zistení z vyšetrovania a odhalení skutočného dôvodu problému nastáva jeho riešenie. To zahŕňa implementovanie zmien prijatých správou zmien, implementovanie náhradných riešení do času implementácie zmeny a hlavne dôkladné otestovanie aplikovaných riešení.

Uzavretie problému
Po konečnej implementácii riešenia a overenia, že riešenie skutočne pokrýva celý problém, dochádza k formálnemu uzavretiu problému. V tomto momente sa môže vytvoriť posudok problému a jeho riešenia. Ten pomôže pri výskyte nových problémov, keď môže poukázať na to, čo sa vykonalo správne a čo nie.

Ukazovatele úspešného zvládnutia správy problémov

Spoločnosť WebStudy kontinuálne vyvíja svoje LMS a pri zavádzaní nových častí systému je ich nedokonalosť očakávaná. Preto len celkový počet vzniknutých problémov nemôže byť ukazovateľom úspešnej správy problémov. To vyžaduje hlbšiu analýzu, ktorá poskytne údaje o jednotlivých častiach systému. Každopádne pokles výskytu incidentov v individuálnych

moduloch a zrýchlenie riešenia problémov môže reflektovať účinnú správu.

Zhrnutie

Úspešné zvládnutie správy problémov má za následok zvýšenie kvality poskytovanej služby. Výkonnosť tohto procesu však závisí na schopnosti personálu, ktorý je do procesu zahrnutý, vo väčšej miere ako pri správe incidentov. Systém napomáhajúci pri správe problémov ovplyvní celý proces v menšej miere, neposkytuje primárne nástroje na zrýchľovanie a zefektívňovanie procesu ako systém pre riadenie incidentov. Ani tak však systém zaoberajúci sa problémami nemôže byť vynechaný z procesov skvalitňujúcich poskytované služby.

2.4 Správa užívateľských požiadaviek na zmenu

Dobre zvládnutá správa problémov a incidentov môže poukázať na nedostatky nachádzajúce sa v poskytovanej službe. Správna identifikácia vzniknutých problémov a implementácia nájdených riešení prispieva k zvýšeniu kvality poskytovanej služby a teda aj k zvýšeniu spokojnosti zákazníka. To však nie je jediná možnosť ako zmeniť službu k prospechu zákazníka. Ďalšou možnosťou je načúvať žiadostiam zákazníka a zmeny, ktoré navrhuje vyhodnotiť a podľa možnosti implementovať. Tento proces nespadá pod prevádzkovanie služby, ale do etapy nasadenia služby do prevádzky.

Podrobnejší opis problematiky správy požiadaviek na zmenu nie je predmetom tejto práce, preto len zhrniem podstatné informácie týkajúce sa neskoršieho návrhu systému a jeho implementácie.

Zber žiadostí o zmenu

Užívateľom je umožnené odovzdávať svoje nápady pomocou jednoduchého formulára, ktorý im napomôže s dodaním potrebných informácií.

Poskytovateľa služby zaujíma predovšetkým náplň požiadavky (čo vlastne užívateľ požaduje), dôležitosť zmeny pre užívateľa a samozrejme informácie o užívateľovi ako také (tj. jeho identifikácia a kontakt). Ak sa užívateľ nevyjadrí dostatočne presne, je mu umožnená komunikácia s tímom zodpovedným za zber žiadostí.

Zaznamenanie a hodnotenie

Všetky žiadosti musia byť zozbierané a záznamy o nich uložené. Následne sú žiadosti vyhodnotené určenou skupinou, zoradené podľa dôležitosti a podľa možností navrhnuté do vývojového procesu. Radenie návrhov sa môže riadiť viacerými kritériami. Môže to byť dôležitosť pre navrhovateľa, ďalej osobná úvaha poskytovateľa služby a taktiež dôležitosť zákazníka.

Uvedenie do vývoja

Schválené návrhy sú následne uvedené do vývoja. Žiadateľ je informovaný o tejto skutočnosti v zázname svojej žiadosti a prípadné nejasnosti sú s nim naďalej odkomunikované.

Kapitola 3

Existujúce riešenia

3.1 Všeobecná charakteristika

Na trhu existuje množstvo riešení systémov poskytujúcich správu incidentov. Tie najkomplexnejšie sa vo svojej podstate diametrálne nelíšia, no aj medzi nimi vieme nájsť rozdiely (riešenie GitHub a BugZilla bude okomentované zvlášť). Vybranými porovnávanými riešeniami sú Zendesk, UserVoice Helpdesk, Freshdesk a Bornevia.

Podstatnou časťou každého komplexného riešenia sú komunikačné kanály. Koncový používateľ je schopný vytvoriť hlásenie pomocou emailu, webového portálu, telefonického hovoru, sociálnych sietí či z mobilnej aplikácie postavenej na kvalitnom API.

Z pohľadu koncového používateľa je podstatné, že všetky systémy sú pre neho jednoduché a prehľadné: dostane sa k svojim hláseniam, k databáze pomocných článkov (knowledgebase), prípadne k fóram.

Knowledgebase je pevnou súčasťou všetkých veľkých systémov. Čiastočne odľahčuje prácu podporného tímu: buď užívatelia nájdu odpoveď na svoj problém v databáze pomocných článkov sami alebo môžu tieto články pri odpovediach použiť agenti podporného tímu.

Ani časť systému pre členov podporného tímu sa vo svojej podstate nelíši. Jednotlivé riešenia však prinášajú svoje vylepšenia, napr. automatizáciu počas života hlásenia, triggery pri zmene alebo vytvorení hlásenia.

Stretávame sa tiež s uľahčením práce podporného tímu. Ten môže odpovedať predpripravenými odpoveďami (tzv. canned answers), pomáhať si s knowledgebase alebo je systém schopný sám analyzovať hlásenie a navrhnúť najlepšie články z knowledgebase.

Systémy tiež umožňujú oddeliť komunikáciu vrámci podporného tímu od komunikácie so zákazníkom.

Všetky analyzované systémy majú tiež pokročilý systém prehľadov o hláseniach. V tomto sa prakticky nelíšia. Sú schopné podať správy o množstvách hlásení a ich stavoch, rozložení prichádzajúcich hlásení v čase, výkonnosti a efektivite práce agentov ako jednotiek i skupín (ak podporujú skupiny). Takisto zaznamenávajú spokojnosť zákazníkov. Nastavenie dohodnutých pravidiel SLA je v porovnávaných riešeniach samozrejmosťou. Napríklad z výkonových správ môžeme vybrať priemerný čas na odpoveď, čas do prvej odpovede, priemerný čas do prvej odpovede.

V ďalších podkapitolách budú vymenované odlišnosti jednotlivých riešení.

3.2 Zendesk

- Návrh odpovedí – analýza hlásení a navrhovanie najlepších článkov z knowledgebase.
- Oddelené zobrazovanie informácií o užívateľovi a jeho organizácie od informácií o hlásení – sprehľadnenie užívateľského prostredia.
- Makrá – umožňuje vytváranie akcií, ktoré sa potom prevedú jedným kliknutím (napr. odpoveď na resetovanie hesla vytvorí správu so správne vygenerovanými informáciami pre daného užívateľa).
- Triggery a automatizácia.
- Zlučovanie hlásení.
- Vytváranie pohľadov na prehľad hlásení.

Zaujímavou technikou je takzvaný otvorený Zendesk. Aplikácia po správnom nastavení umožňuje vloženie hlásenia aj užívateľovi, ktorý sa neverifikoval cez email. Toto riešenie má aj svoje úskalia. Prvé môžeme pozorovať na strane poskytovateľa a tým sú možné nevyžiadané hlásenia – spam. Druhým môže byť strata prehľadu o hlásení na strane užívateľa. Keďže sa neoveril v systéme ako skutočný užívateľ, celý životný cyklus hlásenia prebehne len emailovou komunikáciou. Takto sa mu jednotlivé hlásenia môžu v klientovi stratiť z dohľadu, či na prvý pohľad miešať medzi sebou a trpí tým prehľadnosť. Taktiež to komplikuje identifikáciu užívateľa na strane poskytovateľa.

Treba však poznamenať, že toto je jedna z hlavných požiadaviek na systém na správu incidentov zo strany spoločnosti WebStudy.

3.3 UserVoice

- Všetky informácie na jednej obrazovke (všetky relevantné informácie o užívateľovi sú hneď k dispozícii), mierne neprehľadné.
- Najresponzívnejší dizajn.
- Celkovo menej pokročilých funkcií ako ostatné analyzované riešenia.
- Zlučovanie hlásení.

3.4 Freshdesk

- Kvalitná správa rolí.
- Rozsiahle možnosti fóra.
- LiveChat.
- Zlučovanie hlásení.
- Gamifikácia – odmeňovanie agentov podľa výkonu.

V prospech Freshdesku hovorí hlavne najväčšia flexibilita cenových plánov. Ako jediný z vyššie skúmaných poskytuje aj plán, ktorý je zadarmo. Ten však predpokladá veľkosť tímu do troch členov, čo pre WebStudy nie je vhodné.

3.5 Bornevia

- Najjednoduchšie riešenie zo skúmaných.
- Prehľadný dizajn.
- Nutnosť vlastnej emailovej schránky, z ktorej sa preposiela komunikácia k hláseniam.
- Absencia knowledgebase.
- Koncový užívateľ nemá možnosť prihlásiť sa do systému a prezrieť si svoje hlásenia.
- Neposkytuje telefónny kanál.
- Nízka cena, možnosť nastaviť si cenník priamo podľa požiadaviek.

3.6 BugZilla

BugZilla je otvorený softvér určený na sledovanie chýb pri vývoji projektov. Medzi špecifiká oproti ostatným spomenutým riešeniam patrí, že nie je poskytovaný ako cloudová služba a vyžadoval by teda dedikovaný server, na ktorom by mohol byť prevádzkovaný. To na druhej strane môže byť výhodou, keďže je potom prispôsobiteľný na mieru. Čo už je však väčším problémom, je neschopnosť spolupracovať s databázovým systémom spoločnosti WebStudy. Ak by sme teda uvažovali o použití tohto riešenia, museli by sme vytvoriť API umožňujúce komunikáciu medzi servermi WebStudy a aplikáciou BugZilla. Vzhľadom na technické zameranie BugZilly a jej koncových užívateľov, užívateľské rozhranie je po dizajnovej stránke strohé. Takisto jednotlivé príspevky na časovej osi hlásenia sú síce plne vyhovujúce po obsahovej stránke, no na prvý pohľad neprehľadné pre obyčajného, technicky nie zdatného užívateľa.

3.7 GitHub

GitHub je služba určená na udržovanie zdrojových kódov vyvíjaných softvérových projektov. Jej primárnym cieľom nie je správa incidentov alebo problémov. Ukazuje, ako sa dá navrhnúť systém s presne vymedzenými a postačujúcimi požiadavkami. Poskytuje jednoduchý nástroj, pomocou ktorého jej užívatelia informujú o prípadných chybách konkrétneho projektu. Ďalej umožňuje kategorizáciu jednotlivých hlásení, otvorenú diskusiu prihláseným užívateľom či intuitívne odkazovanie na iné informácie týkajúce sa daného projektu. Užívateľov v tomto prípade nezaujímajú pokročilé štatistiky vzniknutých hlásení a systém ich teda ani neposkytuje.

3.8 Zhodnotenie

Všetky skúmané služby sú vo svojej podstate robustné a komplexné riešenia poskytujúce širokú škálu prostriedkov pre správu. Ak by sme chceli využiť niektoré z nich, je potrebné zamyslieť sa nad viacerými otázkami. Prvou je možnosť integrácie s LMS. Vzhľadom na charakter komunikácie v LMS (len interná komunikácia) by nutné opatrenia pre integráciu mohli jednoducho prevýšiť množstvo práce potrebnej na implementáciu skutočne postačujúceho vlastného systému. Takisto charakteristické požiadavky (napr. vstup do kurzov,

Poskytovateľ	1 polrok	2 polroky	3 polroky	4 polroky
Zendesk	456	912	1368	1824
UserVoice	1080	2160	3240	4320
Freshdesk	384	768	1152	1536
Bornevia	240	480	720	960

Tabuľka 3.1: Ceny poskytovaných služieb (všetky hodnoty v USD)[1]

otvorenie hlásenia z mailového klienta) by vyžadovali náročné zásahy do existujúcej aplikácie. Ďalšou prekážkou môže byť cena. Potrebné prvky služieb (napr. prihlasovanie pomocou tretej strany) môžu cenu navýšiť o takú čiastku, pri ktorej je už výhodnejšie implementovať svoje riešenie. Ceny minimálne postačujúcich plánov pre štyroch členov podporného tímu sú v tabuľke 3.1. Cena riešenia od spoločnosti Bornevia zahŕňa minimálnu konfiguráciu.

Nový systém pre LMS WebStudy sa tak môže inšpirovať už existujúcimi nápadmi. Z týchto riešení si nový systém môže zobrať prehľadné zobrazenie informácií o hlásení a časovú osu hlásenia. Nový systém však nevyžaduje takú komplexnosť, akou disponujú skúmané riešenia.

[1] Zdroj:
https://www.zendesk.com/product/pricing/
https://www.uservoice.com/plans/customer-support/
https://freshdesk.com/pricing
https://www.bornevia.com/pricing

Kapitola 4

Návrh systému

Návrh systému je dôležitou súčasťou vývoja. Dobrý návrh správne identifikuje všetky požiadavky, ktoré sú na systém kladené a pospája ich do súvislostí.

4.1 Vymedzenie požiadaviek na systém

Spoločnosť WebStudy poskytuje službu svojho LMS ako cloud. Znamená to, že jej zákazníci sa môžu k službe dostať kdekoľvek s pripojením do internetovej siete. Ďalšou výhodou je platformová nezávislosť z pohľadu užívateľa: nezáleží na tom, aký typ koncového zariadenia používa, ani aký operačný systém je na danom koncovom zariadení nainštalovaný. Pokiaľ koncové zariadenie obsahuje klienta, ktorý umožní komunikáciu so zdrojmi v cloude, poskytnutiu služby už nič nebráni. Aby sa zaručili tieto možnosti aj po zavedení nového systému na správu incidentov a problémov, musí ich vytvorený systém spĺňať tiež. Najjednoduchším riešením je preto navrhnúť systém ako webovú aplikáciu. Tento prístup taktiež zjednoduší integráciu so zvyškom systému poskytovaného LMS. Systém ako taký musí spĺňať minimálne nasledujúce kritériá:

- otvorenosť – systém musí komunikovať s okolitým svetom, či už sú to užívatelia alebo iné systémy,
- práca v reálnom čase – aby bol systém užitočný bežnému užívateľovi musí s ním priebežne komunikovať,
- bezpečnosť – systém musí zabezpečiť prístup k dátam len pre tých užívateľov, ktorým sú určené.

4.1.1 Definícia požiadaviek daných zameraním systému

Navrhovaný informačný systém má spracovávať incidenty a problémy. Zo zadania vyplýva, že systém musí užívateľom umožniť pracovať s hláseniami počas ich životného cyklu. Úlohy, ktoré túto požiadavku zabezpečia, zahŕňajú:

- vytvoriť hlásenie o incidente alebo probléme,
- umožniť komunikáciu ohľadom hlásenia,
- zmeniť stav hlásenia,
- zmeniť atribúty hlásenia (napr. kategóriu, prioritu, eskaláciu),

- zobraziť existujúce hlásenia,
- priradiť užívateľa k anonymnému hláseniu,
- priradiť člena podporného tímu k hláseniu,
- zobraziť prehľady hlásení.

Jednotlivé úlohy, ktoré systém bude vykonávať, budú špecifickejšie rozčlenené v podkapitole 4.2.

4.1.2 Definícia špecifických požiadaviek

Samotné splnenie požiadaviek spomenutých v podkapitole 4.1.1 však nezaručí, že navrhovaný systém bude vyhovovať spoločnosti WebStudy. Jej pracovníci majú zaužívané isté postupy, s ktorými sú oboznámení aj užívatelia systému.

Systém teda musí oboznamovať užívateľov s aktivitou a zmenami ich hlásení pomocou emailového klienta. Ďalej musí umožniť vstup do časti systému, s ktorou má užívateľ problém, ak je to možné. Rozumie sa tým vstup do kurzov, ak užívateľ špecifikuje, že sa incident vyskytol v ňom. Určitým užívateľom musí sprístupniť aj personálne informácie o užívateľoch, ktorí sú priradení k hláseniu. Nakoniec musí umožniť aj vytvorenie hlásenia užívateľovi, ktorý sa do aplikácie nedokáže prihlásiť.

4.2 Roly užívateľov a prípady použitia

4.2.1 Roly

LMS Webstudy rozoznáva štyri základné užívateľské roly:

- správca systému – pre naše potreby člen podporného tímu (v obrázku 4.1 aktér Support),
- správca inštitúcie (v obrázku 4.1 aktér Administrátor),
- inštruktor,
- študent.

Pre každého zo spomenutých užívateľov má vyvíjaný systém inú množinu povolených prípadov použitia. Vzhľadom na to, že významnejší užívateľ môže uskutočňovať všetky operácie, čo menej významný, vymenovanie možných operácií v systéme pre danú rolu sa obmedzí iba na špecifiká jej prináležiace.

Anonymný užívateľ

Špecifickou rolou je anonymný užívateľ – taký, ktorý sa ešte neprihlásil do systému. Incident sa však môže vyskytnúť v ľubovoľnom bode práce so systémom, a tak užívateľ musí mať možnosť ohlásiť aj situáciu, ktorá vznikne pred jeho prihlásením. Jedinou operáciou, ktorá takémuto užívateľovi bude povolená, je založenie hlásenia o incidente. Potom sa musí spoliehať na to, že ho členovia podporného tímu kontaktujú mimo systém (napr. telefonicky alebo na emailovú adresu). Vytvorenie takéhoto hlásenia vygeneruje a odošle emailovú správu na adresu podporného tímu.

Študent

Študent je chápaný ako prihlásený a overený užívateľ. Keďže si je systém jeho identifikáciou istý, povolí mu aj výpis všetkých hlásení, ktoré sa ho týkajú: buď tie, ktoré sám nahlásil (či už formulárom, telefonicky alebo iným kontaktom s podporným tímom) alebo tie, ku ktorým bol priradený užívateľom s vyššími právami. Ďalej musí mať študent možnosť pridať k hláseniu komentár a hlásenie uzavrieť a tiež uzavreté hlásenie znovu otvoriť. O hlásení má všetky informácie, ktoré zadal, má prístup k všetkým verejným informáciám od členov podporného tímu (správy, zmeny stavu) a vidí aj komunikáciu iných priradených užívateľov.

Inštruktor

Inštruktor sa v ponímaní vyvíjaného systému v zásade nelíši od študenta. Jedinou operáciou, ktorou v oblasti incidentov disponuje naviac, je možnosť priradiť iných užívateľov k hláseniu, ktoré vytvoril. Táto situácia nastáva napríklad vtedy, keď ako majiteľ kurzu spozoruje incident ovplyvňujúci viacerých študentov v danom kurze. Takto má možnosť študentov zainteresovať do incidentu a uľahčí tak voči nim komunikáciu ohľadom riešenia.

Inštruktorom je však umožnené vytvárať hlásenia v oblasti žiadostí o zmenu. Tu si môže prehliadať zaznamenané hlásenia ostatných užívateľov, prispievať svojimi komentármi a hodnotiť požadované zmeny jednoduchým hlasovaním.

Správca inštitúcie

Užívateľ v role správcu inštitúcie má v LMS WebStudy takmer plnú moc nad svojou inštitúciou. To platí aj vo vzťahu k hláseniam o incidentoch spadajúcich do jeho kompetencie. Má možnosť prehliadať si všetky hlásenia, ktoré vytvorili užívatelia jeho inštitúcie alebo ktoré im boli priradené. Taktiež v hlásení vidí informácie, ktoré bežný užívateľ nevidí. Patrí medzi ne zoznam incidentov majiteľa hlásenia, personálne informácie o majiteľovi a priradených užívateľoch a zoznam prihlásení majiteľa do systému LMS. Takisto má možnosť vstupovať do kurzov, ktoré boli špecifikované v hlásení.

Správca systému

Správcami systému sú členovia podporného tímu. Tí majú absolútnu kontrolu nad všetkými hláseniami bez ohľadu na to, do ktorej inštitúcie hlásenia spadajú. Musia mať teda možnosť vidieť zoznam všetkých hlásení. Ďalej môžu založiť hlásenie pre ľubovoľného užívateľa (aj anonymného). Systém im musí umožniť zmenu kategórie, priority i eskalácie hlásenia a taktiež zmenu stavu hlásenia. Ďalej majú možnosť priradiť k hláseniu ľubovoľného člena podporného tímu, ktorý sa hlásením bude zaoberať. Medzi jeho práva patrí aj zmena užívateľa, ktorý hlásenie nahlásil (napr. priradenie anonymného hlásenia) a zmena kurzu, ktorého sa hlásenie týka. Dôležitou funkciou je aj možnosť komunikovať s ostatnými členmi podporného tímu pomocou interných komentárov, ktoré iní užívatelia nevidia. Správcovi systému je tiež umožnené zmeniť zobraziť a zmeniť interné poznámky o jednotlivých užívateľoch.

Špeciálnou kategóriou sú skupiny hlásení. Správcovi systému je povolené vytvárať skupiny hlásení, ktoré majú spoločnú podstatu alebo sú si iným spôsobom blízke. Do a z týchto skupín môže potom hlásenia priradiť alebo odstrániť. Správcovi systému tiež musí byť umožnený prístup k prehľadu o hláseniach a k špecifikácii vlastného prehľadu podľa rôznych atribútov a času.

4.2.2 Prípady použitia

V tejto časti si priblížime špecifické prípady použitia a ich význam v systéme.

Anonymné hlásenia

Ako už bolo vysvetlené, systém musí umožniť anonymný záznam. Tento záznam však môže byť len hlásenie o incidente. Kým správca systému neidentifikuje v systéme užívateľa, ktorý hlásenie vytvoril a nepriradí mu dané hlásenie, je anonymné hlásenie považované za neoverené. To však nebráni tomu, aby sa vzniknutému incidentu popísanému v neoverenom hlásení nevenovala rovnaká pozornosť ako v overenom. Systém tak umožní rovnakú prácu s neovereným hlásením ako s overenými hláseniami. Takéto hlásenie sa však zobrazí iba správcovi systému.

Skupiny hlásení

Mnoho incidentov môže zdieľať rovnakú podstatu alebo sa viažu k odhalenému problému. Keď správca systému hľadá riešenia týchto incidentov, je žiaduce, aby ich mal prehľadne zadelené do skupín. Tieto skupiny môže ľubovoľne vytvárať a meniť. Takto sa znižuje riziko, že pri úspešnom vyriešení problému správca systému zabudne na hlásenia incidentov, ktoré s daným problémom súvisia.

Prehľady hlásení

Prehľady hlásení majú informačný charakter a dvojakú úlohu. Po prvé umožňujú správcovi systému sledovať výkonnosť členov podporného tímu, ako rýchlo odpovedajú užívateľom, za aký čas nájdu riešenia vzniknutých incidentov a tak vytvárajú ukazovateľ dodržania SLA. Po druhé prehľady poukážu na trendy v záznamoch hlásení, vývoj počtu podľa kategórií, eskalácie, priority a pod. Môžu tak prispieť k odhaleniu problému a poukázať na problematické časti systému.

Správca systému musí mať možnosť prehľadne vizualizovať informácie o počtoch hlásení a špecifikovať vlastné obmedzenia pre vytvorenie nového hlásenia – určiť časové obmedzenie, rozčlenenie časového okna na mesiace či týždne, obmedzenie hlásení len so zadanými atribútmi a pod.

Spomenuté prípady použitia sú na obrázku 4.1. Pre zjednodušenie sú roly študenta a inštruktora zahrnuté pod jedného aktéra – užívateľa. Takisto v diagrame vystupuje ďalší aktér – emailový klient. Ten pasívne prijíma a ďalej spracováva správy vygenerované systémom po uskutočnení istých operácií.

4.3 Integrácie komunikačných nástrojov

Aby bol proces hľadania chyby a jej odstraňovania efektívny, je potrebná kontinuálna interakcia s užívateľom, ktorý bol incidentom postihnutý. Tú môžeme docieliť spoľahlivou komunikáciou pomocou rôznych kanálov. Avšak, či už je to komunikácia telefónna, emailová alebo pomocou diskusného vlákna k danému incidentu, užívateľ, ktorý o nových skutočnostiach alebo otázkach na jeho adresu nevie, nemôže ani odpovedať. Je preto nutné užívateľa udržať v povedomí o stave jeho požiadavky či hlásenia. Užívateľ môže byť o zmenách informovaný systémom notifikácií priamo v aplikácii, emailovými správami alebo SMS správami. V danej aplikácii sú najprijateľnejšou formou práve emailové správy smerované do interného

Obr. 4.1: Diagram prípadov použitia

klienta. Po prvé z dôvodu, že zákazník požadujúci SMS integráciu je v menšine, po druhé preto, že systém notifikácií naprieč aplikáciou je v aktuálnom stave nepoužívaný.

4.3.1 Emailový klient

Emailový klient slúži na komunikáciu užívateľov medzi sebou. Je preto prvou, najjednoduchšou a najzreteľnejšou voľbou pri výbere notifikačného kanálu. Náročnejšou voľbou je už výber, ktorú formu emailovej komunikácie využiť. Do úvahy pripadá buď odosielanie

emailov na externé adresy alebo odosielanie emailov v internom klientovi aplikácie. Na prvý pohľad sa zdá ako najlepšou možnosťou skĺbenie oboch postupov, no hlbší pohľad ukazuje viacero úskalí. Ak by sme chceli využiť možnosť odpovede priamo pomocou emailu, rýchlo by sme stratili v kombinovanom prístupe kontrolu nad konzistenciou komunikácie. Jednoducho by nastala situácia, keď má užívateľ pripomienku o správe na dvoch miestach a to môže vyvolať zmätok v tom, z ktorého miesta odpovedať. Aj toto je jedným z dôvodov, prečo zákazníci vzniesli požiadavku na udržovanie komunikácie relevantnej k aplikácii vo vnútri aplikácie. Odpadáva nám teda možnosť externej emailovej komunikácie s užívateľom a ako prostriedok na oboznamovanie užívateľov nám ostáva interný emailový klient. Čo však externej emailovej komunikácii podliehať musí, je oboznámenie členov podporného tímu s novým príchodzím požiadavkom.

Ďalším problémom je rozhodnutie, o čom všetkom má byť užívateľ informovaný. Na jednej strane musí užívateľ vedieť o všetkých dôležitých zmenách v jeho hlásení, na strane druhej nemôže byť danými informáciami zahltený. Najpodstatnejšími zmenami sú vytvorenie hlásenia pre užívateľa pri telefonickom spojení, zverejnenie komentáru k hláseniu a zmena stavu hlásenia. Každá emailová správa tak oznámi užívateľovi zmenu v konkrétnom hlásení. Vzhľadom na využitie interného klienta musí správa obsahovať aj jednoduchý mechanizmus, ktorý umožní prístup ku konkrétnemu hláseniu. Toto riešenie je vyhovujúce aj z dôvodu, že užívateľ je o prijatí emailovej správy informovaný v celej aplikácii.

4.3.2 Systém na správu projektov

Dôležitou súčasťou spoločnosti WebStudy je aj úzka spolupráca s jej zákazníkmi. Snaží sa vyhovieť požiadavkám koncových užívateľov na rôznej úrovni – či už sú to potreby inštitúcií, administrátorov alebo inštruktorov, všetky nápady sú zozbierané a vyhodnotené. Komunikácia ohľadom vylepšení sa však nenachádza priamo v aplikácii, čo môže spôsobovať problémy pri kontaktovaní užívateľov marketingovým tímom. Očakáva sa, že aspoň o niektoré z nápadov spoločnosť prejaví záujem a rozhodne sa pre ich zavedenie do vývojového procesu. Priebeh vývojového procesu je dokumentovaný v systéme na správu projektov a umožňuje vzájomné previazanie položiek odkazmi. Komunikácia ohľadom nápadov, ktoré tak budú novými položkami v systéme, môže byť umiestnená priamo pri nich.

Citrix Podio

Spoločnosť na komunikáciu medzi vývojárskym a marketingovým tímom využíva systém na správu projektov od firmy Citrix, konkrétne produkt Podio. Práve v tomto systéme sú udržiavané poznámky ohľadom vývoja i nápadov a podnetov od užívateľov. Podio poskytuje mechanizmy na spájanie a odkazovanie jednotlivých položiek medzi sebou. Umožňuje to udržanie kontaktu medzi schválenou žiadosťou o vylepšenie a informáciami o vyvíjanej časti aplikácie

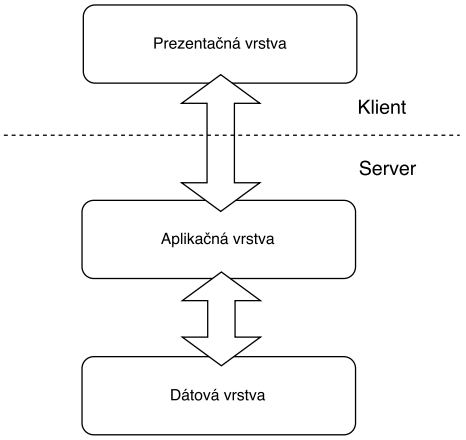
Pre plné porozumenie nápadu niekedy nemusí postačovať prvotná formulácia od užívateľa a ďalšie vysvetlenie je tak nevyhnutné. Preto je ku každej položke potrebná diskusia, a to nielen pôvodného žiadateľa, ale aj ostatných užívateľov. Takto sa tím dozvie, ktoré nápady sú požadované väčším počtom užívateľov. Na to poslúži aj jednoduché hlasovanie o konkrétnych návrhoch.

Všetky návrhy musia mať jednotnú formu. Tím potrebuje identifikovať odosielateľa a udržať si na neho kontakt (emailový alebo telefónny), poznať podstatu návrhu a jeho dôležitosť pre užívateľa. Ďalej je dôležité hodnotenie inými užívateľmi a samozrejme diskusia ohľadom návrhu. Všetky tieto atribúty návrhu tak potrebujeme udržovať v Podiu a takisto

ich dostať do aplikácie. Malý problém však predstavuje to, že Podio neumožňuje vytváranie vlastných typov. To nám znemožní udržovať konkrétne hlasy užívateľov v Podiu a musíme tak tieto hlasy zahrnúť v návrhu databázy.

4.4 Návrh architektúry systému

Navrhovaná časť informačného systému bude poskytovať služby cloudového charakteru, bude to teda webová aplikácia na báze klient-server. Najrozšírenejšou architektúrou takýchto systémov je tzv. viacvrstvová architektúra [1]. Tá rozdeľuje aplikáciu na viacero samostatných vrstiev, ktoré medzi sebou komunikujú. V našom prípade použijeme architektúru trojvrstvovú. Schematické znázornenie vrstiev a smer komunikácie medzi nimi je na obrázku 4.2. Rozhranie medzi klientom a serverom bude jasne definované. Použitou technológiou pre toto rozhranie bude Web API. Musíme teda v serverovej časti aplikácie pevne definovať koncové body, na ktoré bude klientská časť pristupovať, čerpať a odosielať na ne dáta.



Obr. 4.2: Schematické znázornenie trojvrstvovej architektúry[1]

Dátová vrstva

Dátovú vrstvu bude tvoriť databázový systém a bude obstarávať perzistentné uloženie dát. V našom prípade (z dôvodov uvedených v podkapitole 4.3.2) bude táto vrstva distribuovaná medzi viacero zdrojov.

Aplikačná vrstva

Aplikačnou vrstvou rozumieme časť aplikácie zaoberajúcou sa samotnými operáciami s dátami, ich príjem, spracovanie a odoslanie buď do vrstvy dátovej alebo prezentačnej.

Aplikačná vrstva bude navrhnutá za pomoci návrhovej techniky obráteného riadenia (angl. Inversion of control – IoC). To umožní jednoduché oddelenie jednotlivých vyvíjaných častí aplikačnej vrstvy. Uvoľnia sa tak závislosti medzi časťami a môžu byť vyvíjané nezávisle na sebe. Komunikáciu medzi nimi docielime za pomoci vkladania rozhraní do jednotlivých tried, ktoré ich budú implementovať.

Aplikačná vrstva teda bude pozostávať z troch ďalších vrstiev:

[1] Upravené podľa [1].

- **skladová** – hlavnou úlohou je ukladanie a načítanie dát z databázy,
- **servisná** – slúži ako výpočtové jadro, rozhoduje o tom aké dáta sú na základe užívateľských rolí požadované,
- **komunikačná** – zabezpečuje autorizáciu užívateľov pri prístupe na koncové body odhaleného API.

Každú z týchto vrstiev bude predstavovať jedna trieda, ktorej inštancia bude jedináčik (angl. singleton). Návrhový vzor jedináčika využijeme z toho dôvodu, že inštancia triedy na každej vrstve obstará funkcionalitu naprieč celou aplikáciou.

Prezentačná vrstva

Prezentačná vrstva bude jednoznačne oddelená od serverovej časti aplikácie. Bude ju tvoriť tenký klient, ktorého jedinou úlohou tak bude komunikácia s užívateľom pomocou užívateľského rozhrania a príjem a odosielanie dát na server. Tenký sa nazýva preto, lebo sám o sebe nepozná aplikačnú logiku a logicky tak závisí na serveri, ktorý danú logiku implementuje.

Výhod takéhoto oddelenia serveru a klienta je viacero. Prvou je udržovateľnosť a jednoduchý vývoj obidvoch častí. Odhalenie API nám tiež umožňuje vystaviť nad ním viacero klientov, napr. klienta pre mobilné aplikácie. Komunikácia pomocou API je tak platformne nezávislá a pri programovaní nového klienta nevyžaduje masívne zásahy do existujúceho riešenia.

4.5 Návrh serverovej časti systému

4.5.1 Návrh databázy

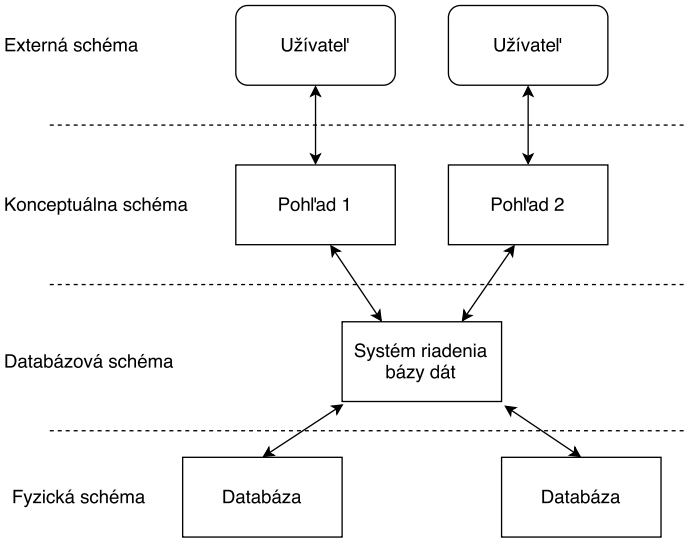
Databázový systém riadi veľké množstvo perzistentných, zdieľaných a spoľahlivých dát [10]. Perzistencia znamená, že dáta sú uchovávané dlhodobo a bez ohľadu na to, či s nimi užívateľ pracuje. Aby boli informačné systémy efektívne, musia umožniť prístup mnohým užívateľom súčasne. Preto hovoríme, že dáta sú aj zdieľané. Ich spoľahlivosť určuje, do akej miery sa môže užívateľ spoľahnúť na ich správnosť, teda na integritu databázy. Tá zaručuje konzistentnosť dát.

Dáta uložené v databázovom systéme musia mať určitú štruktúru. Modelovať túto štruktúru môžeme na viacerých úrovniach. Dnes je najrozšírenejšou technikou štvorúrovňová architektúra modelovania dát (obr. 4.3). Pre potreby návrhu štruktúry dát budeme vychádzať z reálnych požiadaviek na informačný systém. Z nich sme schopní postaviť model s konceptuálnou schémou, ktorý neskôr transformujeme na relačný model databázovej schémy.

Konceptuálny model

Konceptuálny dátový model je založený na grafickej vizualizácii [6]. Dnes sa na tento účel používajú hlavne dve techniky. Po prvé je to diagram tried, po druhé tzv. E-R diagram. V práci bol konceptuálny model vytvorený druhou technikou, teda E-R diagramom.

Aby bol E-R diagram správne použitý, je nevyhnutná znalosť pojmov s ním súvisiacich.



Obr. 4.3: Hierarchické usporiadanie architektúry modelovania

Entita
Reprezentuje typ objektu (objekt záujmu) reálneho sveta. Je popísaná hodnotami svojich vlastností – atribútov.

Atribút
Reprezentuje elementárnu vlastnosť danej entity. Môže byť buď jednoduchý (tvorený nerozdeliteľnou hodnotou) alebo zložený (tvorený viacerými jednoduchými hodnotami – napr. adresa). Poznáme aj odvodené atribúty – také ktorých hodnoty sa dajú odvodiť pomocou operácií nad ďalšími atribútmi.

Vzťah
Vyjadruje väzbu medzi entitami a popisuje ich hierarchiu, závislosť. Vzťah je popísaný tromi charakteristikami:

- stupeň vzťahu – určuje počet entít asociovaných vo vzťahu, vzťah tak môže byť primárny, sekundárny či ternárny,
- kardinalita – určuje počet výskytov entity v jednom konkrétnom výskyte vzťahu,
- voliteľnosť – vyjadruje nutnosť výskytu entity vo vzťahu.

V praxi sa však kardinalita a voliteľnosť zlučujú a vyjadrujú pomocou maximálnej a minimálnej kardinality (napr. 0..n znamená, že vo vzťahu participuje žiadna až mnoho výskytov entity).

Doména
Popisuje množinu hodnôt, z ktorých sa vyberie hodnota atribútu.

Kľúč
Tvorí identifikátor danej entity. Môže byť buď jednoduchý alebo zložený, v závislosti na počte atribútov, ktoré ho tvoria. Ak je daný kľúč unikátny naprieč výskytmi entity, na-

zývame ho kandidátnym kľúčom. Taký kandidátny kľúč, ktorý použijeme na jednoznačnú identifikáciu entity, nazývame primárnym kľúčom.

Pre správnosť konceptuálneho modelu je nutná správna identifikácia entít, ich vzťahov medzi nimi a ich atribútov. Z požiadaviek na systém vieme identifikovať tieto entity:

- hlásenie,
- správa,
- záznam o zmene stavu,
- záznam o zmene atribútov (kategória, eskalácia, priorita),
- záznam o zmene priradených užívateľov,
- skupina hlásení,
- hlasovanie o vylepšení,
- užívateľ,
- kurz,
- hlásenie v Podiu.

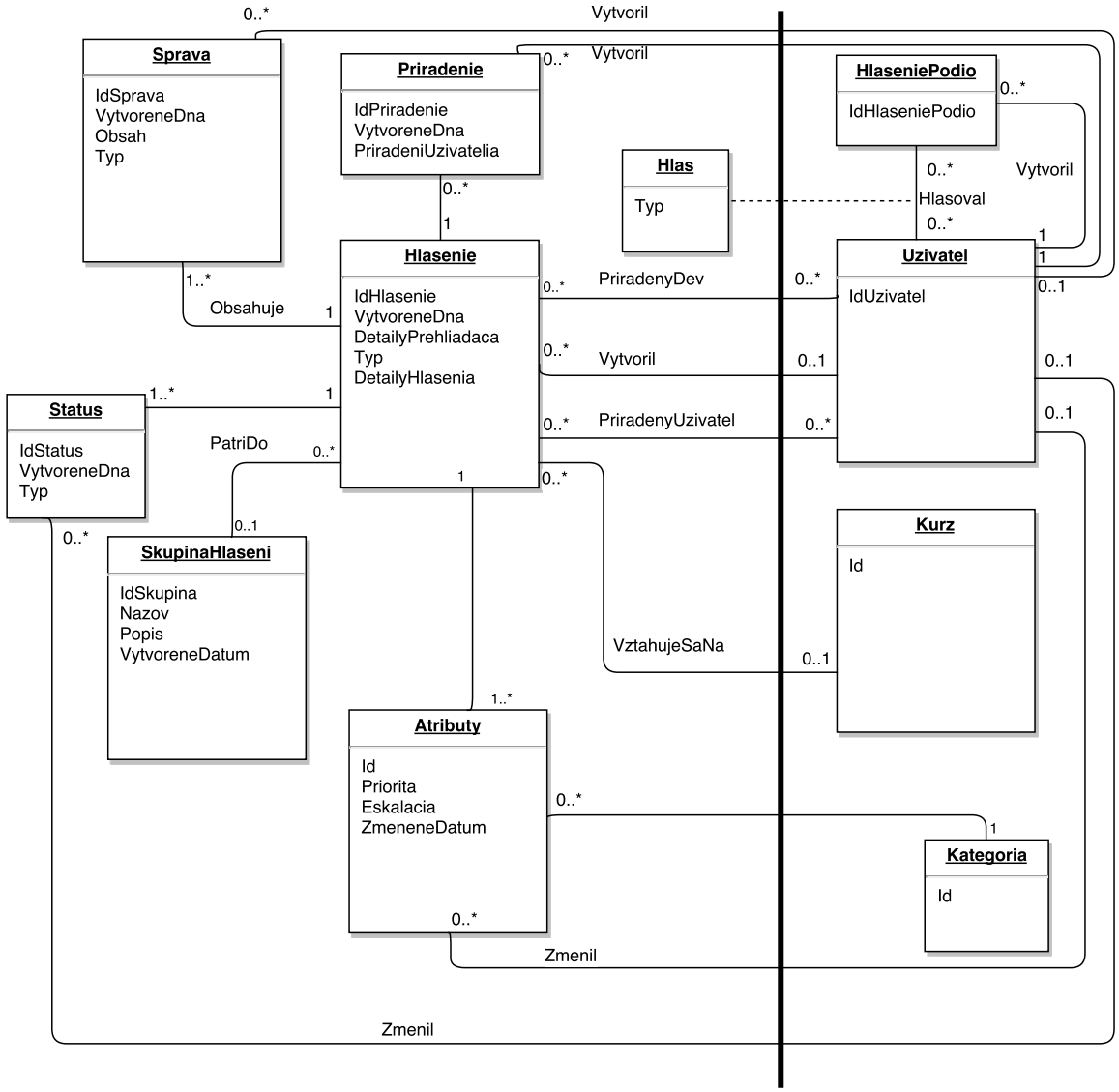
Atribúty týchto entít a ich vzťahy medzi nimi sú znázornené na obrázku 4.4. Za zmienku stojí popis vzťahov entity *Hlasenie* s ostatnými entitami a vyznačenie entít už existujúcich (*Uzivatel*, *Kurz*, *Kategoria*) alebo stojacich mimo implementovanú databázu (*HlaseniePodio*).

Najprv sa pozrieme na vzťahy entity *Hlasenie*. Z obrázku je jasné, že každá *Sprava*, *Stav*, *Atributy* a *Priradenie* patrí práve jednému výskytu *Hlasenia*. *Hlasenie* môže, ale nemusí patriť nanajvýš jednému výskytu *SkupinaHlaseni*. Aby sme vyhoveli požiadavke anonymného prispievania k hláseniam, tak žiadny výskyt entity *Sprava*, *Stav*, *Atributy* alebo *Hlasenie* nemá povinnosť vystupovať vo vzťahu s užívateľom. Neskoršiu identifikáciu užívateľa, ktorému bude hlásenie patriť potom určí zložený atribút *DetailyHlasenia*. Čo sa však týka vzťahu *Vytvoril* medzi entitami *HlaseniePodio* a *Uzivatel*, kardinalita na strane *Uzivatela* je prave jedna. To zabezpečí manipuláciu s výskytmi entity *HlaseniePodio* len prihlásenému užívateľovi. Ostatné vzťahy sú zreteľné z obrázku 4.4. V ňom je hrubou čiarou oddelená časť systému, ktorá bude implementovaná priamo v systéme LMS. Vpravo od čiary sa nachádzajú už existujúce entity (*Uzivatel*, *Kurz*, *Kategoria*) a entita, ktorej výskyty budú udržované mimo systém LMS v Podiu (*HlaseniePodio*).

4.5.2 Návrh API

Komunikačné rozhranie medzi klientom a serverom poskytuje jediný prístup klienta do logickej časti systému. Je preto potrebné, aby všetky požiadavky boli obslúžiteľné pomocou koncových bodov API. Dá sa teda povedať, že pre každý prípad použitia existuje jeden koncový bod API. Je vhodné vytvoriť rôzne koncové body aj pre operácie pracujúce s rovnakou formou dát, ale s trochu rozličným obsahom (napr. pre interné a externé správy), keďže na tento obsah nemusí mať každý užívateľ právo. Zjednoduší sa tiež kontrola autorizácie a rozšíria sa možnosti záznamu aktivity užívateľov v systéme. Čím viac je špecifických akcií, tým podrobnejšie sa dá užívateľovo správanie a jeho akcie zaznamenať.

Z tohoto dôvodu bude API poskytovať samostatný prístupový bod pre každú užívateľskú akciu.



Obr. 4.4: Konceptuálny model dát

4.6 Návrh klientskej časti systému

Návrh klientskej časti systému je dôležitou súčasťou vývoja aplikácie. Keďže klient je jedinou možnosťou, ktorú užívatelia majú pre prácu so systémom, musí sprístupňovať všetky funkcie im prináležiace. Na to aby užívateľ so systémom pracoval efektívne, klient musí byť prehľadný a navigácia v ňom plynulá. Prehľadnosť je ovplyvnená množstvom potrebných informácii, ktoré je nutné zobraziť. Vzhľadom an fakt, že rozdielne cieľové skupiny využívajú odlišné sady informácií, musíme sa zamerať aj na to, čo ktorá skupina požaduje.

4.6.1 Informácie o cieľovej skupine

Systém pre správu incidentov a problémov je primárne určený pre členov podporného tímu. Na to, aby tento personál vykonával svoju prácu na požadovanej úrovni, musí byť dostatočne technicky zdatný. Takémuto užívateľovi teda nebudú vytvárať problém zložitejšie postupy pri vykonaní požadovaných operácií. Taktiež bude prichádzať do kontaktu so systém často a dané náročnejšie operácie preto po zžití sa so systémom nebudú mať významný negatívny dopad na jeho prácu. Z toho vyplýva, že obrazovky, ktorými bude disponovať tento užívateľ nemusia byť maximálne zjednodušované, skôr by mali obsahovať všetky ovládacie prvky. Podobná charakteristika ako pre členov podporného tímu platí aj pre administrátorov inštitúcií.

Členovia podporného tímu a administrátori nebudú jedinými užívateľmi, ktorí do styku so systémom prídu. Definovať a správne odhadnúť zvyšok cieľovej skupiny však môže byť náročné. Spoločnosť WebStudy poskytuje svoj systém LMS rozličným inštitúciám. Počet koncových užívateľov systému sa pohybuje v desiatkach tisíc a jednoznačná charakteristika užívateľa je teda nemožná.

Aké informácie však k dispozícii sú, je približná charakteristika užívateľov, ktorí sa obrátili na členov podporného tímu telefonicky v minulosti. Zväčša to boli technicky menej zdatní užívatelia, nevyužívajúci prenosné zariadenia na prácu so systémom. Je jasné, že práve na užívateľov, ktorí sa na podporný tím obracajú musí byť braný ohľad pri návrhu a tvorbe užívateľského rozhrania. Práve užívateľom najviac využívajúcim navrhovaný systém musí užívateľské rozhranie vyhovovať.

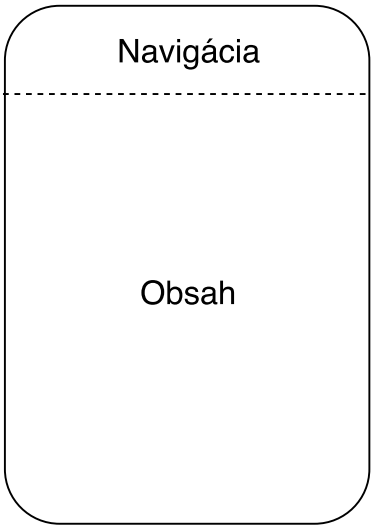
4.6.2 Návrh užívateľského rozhrania

Návrh systému počíta s rozčlenením cieľovej skupiny podľa užívateľských rolí. Je teda žiaduce pripraviť užívateľské prostredie konkrétnemu užívateľovi z cieľovej skupiny.

Pre všetky zobrazované stránky platí, že rozloženie obsahu musí byť prehľadné. Mriežkové rozloženie stránky užívateľom poskytne vizuálne vedenie po stránke a umožní im nájsť obsah i navigáciu bez zbytočného hľadania [4]. Stránka teda bude rozdelená na dve časti – časť navigačnú a časť obsahovú (obrázok 4.5).

Podstatnou sekciou bude prehľad hlásení. Tá musí umožňovať členom podporného tímu rozdelenie hlásení na problémy a incidenty. Ďalej musí pre všetkých užívateľov poskytovať informácie na jednoznačnú identifikáciu hlásenia. Tou nie je len číslo hlásenia, ale aj kto hlásenie podal, záznam o poslednej aktivite a atribútoch hlásenia. Táto sekcia poskytne tiež možnosť rýchlej navigácie k podaniu nových hlásení.

Podávanie nových hlásení pozostáva z dvoch častí. Tou prvou bude formulár na vyplnenie všetkých požadovaných informácií. Táto časť je viditeľná pre inštruktorov, študentov i administrátorov. Druhú časť budú tvoriť podrobné informácie o užívateľovi, ktorého sa dané hlásenie týka. Túto sekciu uvidí iba člen podporného tímu.



Obr. 4.5: Zjednodušené rozvrhnutie stránky podľa mriežky

Detail hlásenia bude veľmi podobne rozdelený na dve časti. V prvej sa nachádza prehľadné zobrazenie životného cyklu hlásenia, ktoré uvidia všetci používatelia, v druhej znova súhrn informácií o danom užívateľovi.

Poslednou významnou sekciou je zobrazenie prehľadov hlásení. Táto časť je určená výhradne členom podporného tímu. Prináša informácie o stavoch incidentov, ich kategóriách a rôznych atribútoch. Hlavným bodom je teda vizualizácia týchto informácii pre potreby porovnania medzi sebou. Toto porovnanie je najlepšie predvedené v grafickej podobe a bude teda využívať grafy. Vzhľadom na množstvo informácií bude zobrazovaná v grafe vždy len jedna podstatná informácia, podľa ktorej môžeme porovnať jednotlivé prehľady. Všetky dostupné informácie sa budú nachádzať aj v textovej podobe pod grafom.

Dôležitou požiadavkou je responzívnosť užívateľského rozhrania. Aplikácia musí umožňovať užívateľom prácu aj na prenosných zariadeniach. Toto však platí iba pre študentov a inštruktorov, ktorí na prístup k aplikácii využívajú mobilné zariadenia najčastejšie. Pre administratívne účely sa malé displeje týchto zariadení nehodia a rozhranie pre administrátora a člena podporného tímu tak nemusí byť optimalizované pre malé rozlíšenia obrazoviek.

Kapitola 5

Implementácia navrhnutého systému

5.1 Voľba implementačných prostriedkov

Voľba implementačných prostriedkov bola uľahčená a obmedzená existujúcim systémom LMS WebStudy. Pre serverovú časť systému sú to tieto:

- Microsoft SQL Server – pre dátovú vrstvu,
- .Net 4, jazyk C# – pre servisnú a komunikačnú vrstvu.

Klientská časť aplikácie bude implementovaná pomocou aplikačného rámca AngularJS. Ten využíva technológiu JavaScript, HTML5 a CSS3. AngularJS umožňuje vytváranie tzv. single-page webových aplikácií. Silnými stránkami tohto aplikačného rámca je AJAX komunikácia a vytváranie tzv. CRUD (z angl. Create, Read, Update, Delete) typ aplikácií [7]. Technológia AJAX (Asynchronous JavaScript and XML) umožňuje vytváranie aplikácií, s ktorými môže užívateľ pracovať flexibilnejšie, keďže sa mu nenačítava celá stránka pri každom kliknutí. Navigácia aplikáciou je vedená pomocou smerovania. Vybraná cesta (route) aplikáciou je určená tvarom URL adresy. Každá URL adresa tak predstavuje jedinečnú cestu, ktorá má svoju šablónu a nad ktorou AngularJS umožňuje plnú kontrolu pomocou rôznych komponentov. Podrobnejšie sa o implementácii klientskej časti zmienime v podkapitole 5.3.

Ďalšie podporné knižnice využité pri implementácii systému sú uvedené v prílohe A.

5.2 Implementácia serverovej časti

Databázový model

Najnižšou vrstvou implementovaného systému je dátová vrstva, ktorú tvorí databáza a všetky dáta v nej. Tie sú istým spôsobom štrukturované. V tejto podkapitole objasníme prevod konceptuálneho modelu dát do relačného databázového modelu.

Pod pojmom relácia budeme rozumieť dvojrozmernú dátovú štruktúru pozostávajúcu zo záhlavia relácie a tela relácie. Nech $D_1, D_2, \ldots D_n$ sú množiny atomických hodnôt – domény a $A_1, A_2, \ldots A_n$ sú názvy atribútov. Záhlavím relácie je množina dvojíc (A_i, D_i), kde sú všetky A_i unikátne. $R \subseteq D_1 \times D_2 \times \cdots \times D_n$ tvorí telo relácie [3].

Zjednodušene tomu môžeme rozumieť tak, že relácia je tabuľka s pevne danou schémou (záhlavie tabuľky) – zoznamom atribútov. Telo relácie tak tvoria riadky danej tabuľky.

Ak chceme previesť jednotlivé entity z konceptuálneho modelu na tabuľky v relačnom modeli, riadime sa nasledujúcim: každú entitu bude predstavovať jedna tabuľka, schému tabuľky budú tvoriť atribúty entity a zložené atribúty transformujeme na jednoduché ich rozložením.

Podrobný popis všetkých schém vzniknutých relácií nie je významný a v práci sa ním nebudeme zaoberať.

Transformácia vzťahov je zložitejšia. Závisí na kardinalite vzťahu medzi entitami. Ak je vzťah kardinality *jedna k jednej*, zvolíme jednu z tabuliek (je jedno ktorú) a vložíme do nej novú položku – cudzí kľúč, ktorý bude odkazovať na konkrétny riadok druhej tabuľky. Ak je vzťah kardinality *jedna k mnoho*, tak zvolíme tabuľku reprezentujúcu entitu, ktorej výskyt vo vzťahu je viacnásobný a vložíme do nej znova cudzí kľúč odkazujúci sa do druhej tabuľky. Transformácia vzťahu *mnoho k mnoho* je najzložitejšia. Vyžaduje si novú väzobnú tabuľku, v ktorej bude primárnym kľúčom dvojica cudzích kľúčov odkazujúcich sa do tabuliek vstupujúcich do vzťahu.

Takúto tabuľku budú vyžadovať vzťahy priradených užívateľov a priradených členov podporného tímu.

Transformovaný konceptuálny model je zobrazený v prílohe B. Tento diagram ukazuje reálny obraz databázových tabuliek a všetkých podstatných vzťahov.

Samotné tabuľky a vzťahy medzi nimi boli vytvorené pomocou nástroja Microsoft SQL Server Management Studio 2012 a jeho grafického rozhrania. Vytvorenie tabuliek zahŕňalo aj definovanie cudzích kľúčov v tabuľkách, označenie primárnych kľúčov a povinných položiek a nastavenie východzích hodnôt pre tieto položky. Sú to všetky časové známky vytvorených záznamov v tabuľkách.

Logika aplikácie

Ďalšou vrstvou systému je vrstva aplikačná, ktorá zabezpečuje logické jadro aplikácie. Manipuluje s dátami z dátovej vrstvy a triedi informácie prichádzajúce z prezentačnej vrstvy.

Aplikačná vrstva (ako bolo spomenuté v podkapitole 4.4), pozostáva z troch vrstiev.

Vrstvu skladovú predstavuje trieda `IssueRepository` (definovaná v súbore `IssueRepository.cs`). Implementuje rozhranie `IIssueRepository`. Model dát, s ktorými objekt pracuje, je definovaný v súbore `WSIssue.cs`. Komunikáciu s databázou uľahčuje `EntityFramework`, ktorý má za úlohu vytvárať databázový koncept a spolu s integrovaným jazykom LINQ zjednodušujú tvorenie databázových dotazov. Zaujímavou časťou skladovej vrstvy je vytváranie prehľadov o hláseniach. Na základe užívateľom zadefinovaných podmienok (dĺžka časového obdobia a prípadné rozdelenie na mesiace alebo týždne) sa vytvoria časové rámce, v ktorých sa sledujú stav a atribúty jednotlivých hlásení. Pre každý vytvorený časový interval sú potom zozbierané jednoduchšie prehľady na základe ďalších podmienok. Užívateľ môže obmedziť prehľad na žiadnu, niektoré alebo všetky možnosti z priorít, eskalácie, kategórií, stavu hlásenia či podľa toho, kto vytvoril hlásenie. Taktiež sa zozbierajú priemerné časy do prvej odpovede na hlásenie a taktiež časový úsek, ktorý zabralo vyriešenie hlásenia. Každý časový úsek predstavuje teda súbor kolekcií čiastkových prehľadov, priemerný čas odozvy a priemerný čas vyriešenia hlásenia.

Trieda reprezentujúca prehľad o hláseniach má nasledujúcu štruktúru:

```
public class WSIssueReport
  {
    public WSIssueInterval interval { get; set; }
    public ICollection<WSIssueReportItem> categories { get; set; }
    public ICollection<WSIssueReportItem> types { get; set; }
    public ICollection<WSIssueReportItem> status { get; set; }
    public ICollection<WSIssueReportItem> tier { get; set; }
    public ICollection<WSIssueReportItem> priority { get; set; }
    public ICollection<WSIssueReportItem> creators { get; set; }
    public int responseTime { get; set; }
    public int resolveTime { get; set; }
  }
```

Jednotlivé čiastkové prehľady sú tvorené názvom hodnoty, podľa ktorej sa hlásenia filtrovali a počtom hlásení, ktoré danému filtrovaniu vyhovujú.

```
  public class WSIssueReportItem
  {
    public string name { get; set; }
    public int count { get; set; }
  }
```

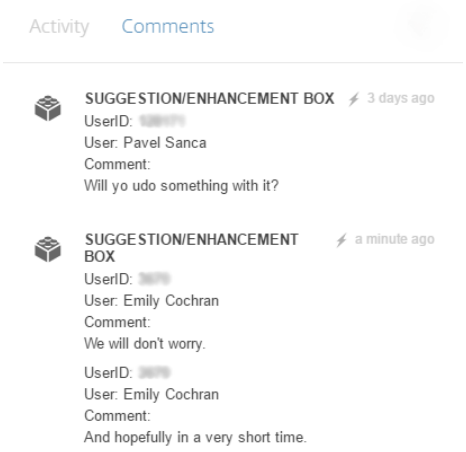
Toto riešenie umožňuje jednoduchú vizualizáciu poskytnutých dát v klientskej časti, či v textovej podobe alebo vo forme grafov.

Servisnú vrstvu vytvára trieda `IssueService`. Jej úlohou je vyskladať informácie z viacerých zdrojov na základe prichádzajúcej požiadavky. Udržuje si tiež povedomie o aktuálne prihlásenom užívateľovi. Vďaka tomu môže rozhodovať o rozsahu informácií, ktoré užívateľ obdrží. Príkladom môže byť priorita a eskalácia požadovaného hlásenia, pri ktorých je žiaduce, aby ich videl iba člen podporného tímu. Potrebný model dát, s ktorým vrstva pracuje, je definovaný v súbore `ViewIssue.cs`.

Ďalšou dôležitou úlohou servisnej vrstvy je spojenie so vzdialeným systémom PMS Podio. Pomocou klienta Podio.NET vytvárajúcim rozhranie medzi API Podia a aplikáciami na platforme .NET je umožnená komunikácia len pomocou volania metód nad inštanciou objektu `Podio`. Poskytnutý klient je však dvojaký: umožňuje komunikáciu synchrónnu aj asynchrónnu. Vzhľadom na fakt, že servisná vrstva v tomto prípade tvorí len medzičlánok medzi Podiom a klientskou časťou aplikácie a obidvoma smermi musí odpovedať dátami, ktoré sama nemá k dispozícii, je zvolený klient synchrónny.

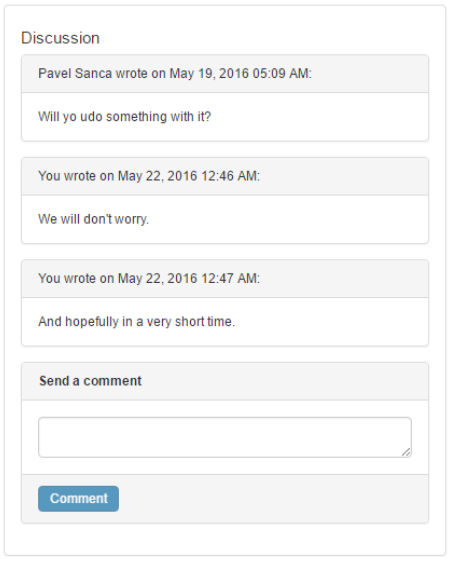
Prvým krokom k integrácii Podia je vytvorenie aplikácie v systéme Podio. Použitím webového rozhrania si vytvoríme v aplikácii šablónu, ktorá bude korešpondovať s položkami v systéme LMS WebStudy. Ďalším krokom je rozhodnutie o forme autentizácie. Buď sa pri požiadavkách smerom k Podiu bude autentizovať užívateľ alebo aplikácia. Vzhľadom na fakt, že užívatelia účet v Podiu nemajú a prakticky o použití Podia nevedia, autentizovať sa LMS bude pomocou aplikácie samotnej. To síce umožní iba prístup ku konkrétnej aplikácii, no iné dáta z Podia ani LMS nepotrebuje. Po úspešnej autentizácii už môže prebehnúť samotná výmena dát.

Zaujímavý problém predstavuje implementovanie komentárov do Podia. Šablóna položky aplikácie nedovoľuje vytvoriť pole komentárov ani komentovať položky neautentizovaným užívateľom, a preto si systém musí pomôcť inak. To, čo Podio dovoľuje, je pridávanie komentárov autentizovanej aplikácii. Keďže už aplikáciu systém LMS má, vytvorí komentár, ktorého autorom bude aplikácia v Podiu (viď obrázok 5.1).



Obr. 5.1: Ukážka komentárov vytvorených v LMS a zobrazených v Podiu

Komentár bude mať pevnú štruktúru, aby sa dal spätne identifikovať užívateľ LMS a aby bol tento komentár odlíšiteľný od komentárov interných – tých, ktoré vytvoria členovia podporného tímu priamo v Podiu. Bude obsahovať povinné položky *UserID*, *User* a *Comment*. Po prijatí všetkých komentárov systém vyhodnotí ako zobraziteľné iba tie komentáre, ktoré budú zodpovedať danej štruktúre. Na obrázku 5.2 môžeme vidieť úspešnú spätnú identifikáciu užívateľa, ktorý komentár vytvoril.



Obr. 5.2: Ukážka komentárov v LMS

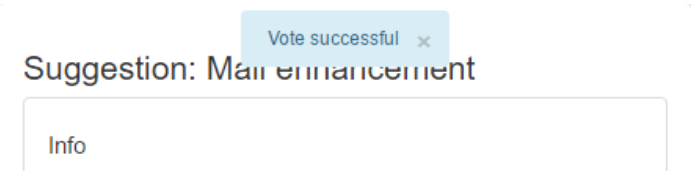
Poslednou vrstvou serverovej aplikačnej vrstvy je vrstva komunikačná. Tá je reprezen-

tovaná triedou `IssuesController` a konfiguráciou koncových bodov Web API. V triede `IssuesController` sú definované a implementované funkcie obstaravajúce jednotlivé zaregistrované koncové body. Pri registrácii koncových bodov je nutné špecifikovať kontrolér a akciu, ktorú má byť vykonaná. Touto akciou je jedna z už spomenutých funkcií kontroléru. Vrstva zodpovedá za serializáciu a deserializáciu odchádzajúcich a prichádzajúcich dát.

5.3 Implementácia klientskej časti

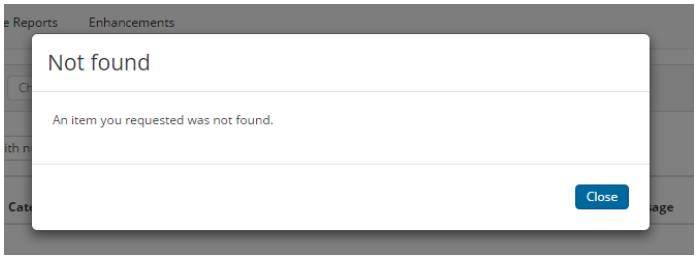
Klient ako taký musí umožňovať užívateľovi navigáciu naprieč systémom. Navigácia v single-page aplikáciách by mohla byť problematická (keďže sa jedná o jednu stránku), a tak aplikačný rámec AngularJS (ďalej Angular) poskytuje možnosť upravenia URL adresy, aby odpovedala tomu kde sa užívateľ nachádza a umožnila mu poznamenať si to. Jednotlivým URL adresám priraďuje kontroléry (spravujúce logiku aplikácie) a šablóny (na vizualizáciu dát). Aplikačný rámec UI-Router rozširuje možnosti smerovania samotného Angularu a umožňuje vytváranie hierarchického stromu stavov. Dodáva stavový automat, ktorý spravuje navigáciu aplikáciou a poskytuje tiež vnorené stavy a komunikáciu medzi nimi.

Vyvíjaná aplikácia preto bude rozdelená na stavy. Budú to prehľady hlásení, prehľady vylepšení, detail hlásenia, detail vylepšenia a stavy pre odoslanie hlásenia i vylepšenia. Každý stav si po vyvolaní najprv načíta potrebné dáta z API servera. Táto časť nám zabezpečí autorizáciu užívateľa voči stavu. Akonáhle si užívateľ vyžiada dáta ku ktorým nemá prístup, zo serveru príde zamietavá odpoveď (kód odpovede 401). To sa však pri bežnej navigácii aplikáciou nestane – aplikácia nesmie poskytnúť užívateľovi možnosť v menu, ktorá mu je neprístupná. Čo však nemôže aplikácia priamo ošetriť, je manuálne zadanie URL cesty reprezentujúcej stav, ku ktorému by užívateľ nemal mať prístup. V tomto bode sa musí klientská časť spoľahnúť na serverovú časť, ktorá užívateľa autorizuje. Keď má užívateľ všetky potrebné dáta, inicializuje sa potrebný stav a zobrazia sa relevantné údaje. Vzhľadom na to, že všetka komunikácia je asynchrónna, a prehliadač nemá dohľad nad načítaním stavov, užívateľ musí byť informovaný o každom čakaní na dáta i o každej chybe, ktorá sa mohla vyskytnúť na serveri. To je dosiahnuté pomocou modálových okien (príklad na obrázkoch 5.3 a 5.4).



Obr. 5.3: Ukážka oznámenia o úspešnej akcii

Špecifickou integráciou je reCAPTCHA od spoločnosti Google. Vzhľadom na fakt, že do aplikácie má prístup aj anonymný užívateľ, konkrétne do stavu odoslania hlásenia, nie je v tejto časti žiadna autorizácia. Tento stav teda nemá práva žiadať o akékoľvek dáta zo servera, no musí nejaké odosielať. Aby sa zabránilo zneužitiu tohoto stavu automatizovanými službami, je súčasťou formulára aj overenie užívateľa pomocou reCAPTCHA. Keďže je tento stav zdieľaný aj pre užívateľov prihlásených do systému, za pomoci mechanizmov poskytovaných Angularom je reCAPTCHA z formulára pre týchto užívateľov odstránená.



Obr. 5.4: Ukážka oznámenia o chybe

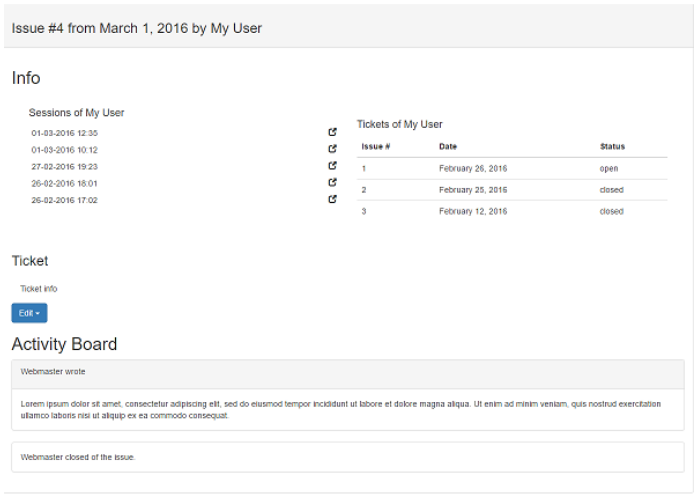
5.4 Testovanie

Testovanie systému je dôležitou súčasťou vývoja a zavedenia systému do prevádzky. Môže včas odhaliť chyby, ktoré by neskôr mohli spôsobiť veľké komplikácie pri ich odstraňovaní. Takisto môže poukázať na slabšie miesta návrhu či už aplikačnej logiky alebo užívateľského rozhrania. V tejto práci je testovanie zamerané práve na užívateľa a jeho požiadavky.

Testovanie užívateľského rozhrania

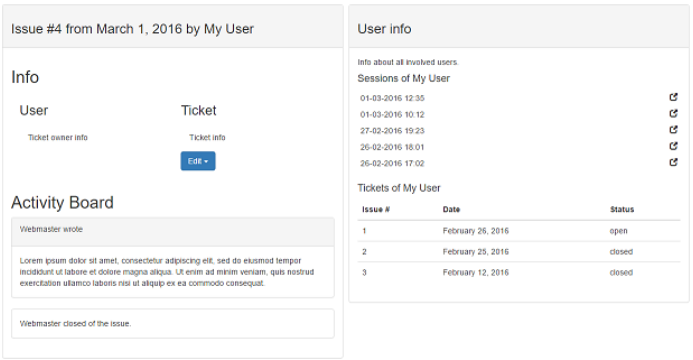
Vzhľadom na to, že systém má zjednodušiť a sprehľadniť správu incidentov a problémov, musí tomu odpovedať aj užívateľské rozhranie. Hlavným používateľom systému sú členovia podpory a práve na ich vzorke prebehlo užívateľské testovanie. Počas vývoja im boli poskytnuté prototypy užívateľského rozhrania a bola sledovaná ich práca a navigácia v tomto rozhraní. Keďže hlavnou požiadavkou návrhu užívateľského rozhrania bola prehľadnosť a dostupnosť ovládacích prvkov, užívatelia už pri prvom kontakte so systémom zvládali jeho obsluhu na slušnej úrovni. Avšak iba návrh nemôže vyriešiť všetky nejasnosti.

Prvou pozorovanou nevyhovujúcou skutočnosťou bolo rozloženie stránky detailu hlásenia v administrátorskom pohľade. Prvý návrh počítal s tým, že všetky informácie ohľadom hlásenia budú dostupné na začiatku detailu. Obidve sekcie informácií o užívateľovi aj o hlásení však zaberali na obrazovke priveľa miesta a keď chcel užívateľ študovať komunikáciu nachádzajúcu sa pod nimi musel až pričasto posúvať stránku zhora nadol (obrázok 5.5).



Obr. 5.5: Návrh administrátorského rozhrania č. 1

Riešením tohto problému bolo teda rozdelenie obrazovky vertikálne na dve polovice. V jednej sa nachádzajú informácie o hlásení spolu s komunikáciou, v druhej kompletné informácie o užívateľoch, ktorých sa hlásenie týka (obrázok 5.6).



Obr. 5.6: Návrh administrátorského rozhrania č. 2

Takto odpadol problém s pričastým posúvaním stránky, čo bolo dokázané v ďalšej iterácii vývoja prototypov. S tým však vyvstal problém nový. Rozdelenie obrazovky vertikálne znemožňuje praktické použitie administrátorského režimu na mobilných zariadeniach. Treba však povedať, že využitie mobilných zariadení medzi členmi podporného tímu pre účely administrácie systému je prakticky nulové.

Ďalším pozorovaným nedostatkom pôvodného návrhu bola zmena atribútov hlásenia, konkrétne uloženie ovládacích prvkov, ktoré za ne zodpovedajú. Prvotnou myšlienkou bolo, aby pri kliknutí na atribút označený symbolom, bol daný atribút zmenený. Táto funkcionalita však bola prehliadaná prakticky všetkými užívateľmi. Rozhodlo sa preto, že pre zvýšenie prehľadnosti budú všetky prvky meniace atribúty hlásenia zaradené pod tlačidlom Upraviť (Edit). Táto možnosť hneď na prvý pohľad hovorí, čo sa pod danou funkciou skrýva.

Kapitola 6

Záver

Udržanie kvality prevádzkovanej služby na požadovanej úrovni vyžaduje plné nasadenie členov podporného tímu. Ich prácu v mnohom zefektívňujú a podporujú nástroje, ktoré sú im poskytnuté. Spokojnosť zákazníka sa odvíja nie len od bezchybného behu požadovanej služby, ale aj od schopnosti podporného tímu rýchlo a účelovo riešiť prekážky, ktoré vzniknú počas prevádzkovania služby.

Cieľom tejto bakalárskej práce bolo nahliadnuť do problematiky správy incidentov a problémov a navrhnúť systém, ktorý bude spomenuté procesy podporovať. Porovnanie existujúcich riešení napomohlo k špecifikácii požiadaviek na implementovaný systém. Daný informačný systém bol vyvíjaný v spolupráci so spoločnosťou WebStudy a integrovaný do jej produktu LMS WebStudy. Jeho návrh teda sledoval nielen naplnenie požiadaviek vyplývajúcich zo zamerania systému, ale aj jednoduchú integráciu s ostatnými časťami LMS.

Implementovaný systém sa z dôvodu časovej kolízie naplánovaných úloh nepodarilo dostať do skúšobnej prevádzky so vzorkou koncových užívateľov, ale členovia podporného tímu sa s ním začali zoznamovať.

Návrh tohto systému podporoval jeho jednoduchú rozšíriteľnosť. Umožňuje obohatenie systému o ďalšie časti, ktoré by podporovali prevádzku služby poskytovaného LMS, napr. spĺňanie užívateľských požiadaviek. Ešte dôležitejším rozšírením systému bude neskoršia integrácia s databázou znalostí spoločnosti WebStudy. Nasadenie tejto časti systému do prevádzky urýchli prácu členov podporného tímu a odľahčí ho od nemalého množstva požiadaviek.

Literatúra

[1] *Architektury informačních systémů*. Prezentácia prezentovaná na: [Informačné systémy, FIT VUT Brno].

[2] Informace. In: *Česká terminologická databáze knihovnictví a informační vědy (TDKIV)* [online]. Praha: Národní knihovna ČR, 2003 [cit. 12.5.2016]. URL http://aleph.nkp.cz/F/?func=direct&doc_number=000000456&local_base=KTD

[3] Chochlík, M.; Matiaško, K.; Vajsová, M.; aj.: *Databázové systémy: základy databázových systémov*. Žilina: Žilinská univerzita v Žiline, 2008, ISBN 978-80-8070-820-7.

[4] van Duyne, D. K.; Landay, J. A.; Hong, J. I.: *Návrh a tvorba webů. Vytváříme zákaznícky orientovaný web*. Brno: CP Books, 2005, ISBN 80-251-0508-3.

[5] Hruška, T.; Křivka, Z.: *Pojem informačního systému – Data – Procesy – Transakce. Studijní opora*. FIT VUT v Brně, 2012.

[6] Kaluža, J.; Kalužová, L.: *Modelování dat v informačních systémech*. Praha: Ekopress, 2012, ISBN 978-80-86929-81-1.

[7] Kozlowski, P.; Darwin, P. B.: *Mastering Web Application Development with AngularJS*. Birmingham: Packt Publishing, 2013, ISBN 978-1-78216-182-0.

[8] OGC: *ITIL: Service Operation*. Londýn: TSO, druhé vydání, 2011, ISBN 9780113313075.

[9] OMNICOM: *Co je to služba IT?* [online]. Vytvorené 2008 [cit. 12.5.2016]. URL https://www.bestpractice.cz/cs/Best-practice/-ITSM-ITIL-/-Co-je-to-sluzba-IT.alej

[10] Pokorný, J.; Valenta, M.: *Databázové systémy*. Praha: České vysoké učení technické v Praze, 2013, ISBN 978-80-01-05212-9.

Prílohy

Príloha A

Zoznam použitých knižníc tretích strán

A.1 Serverová časť aplikácie

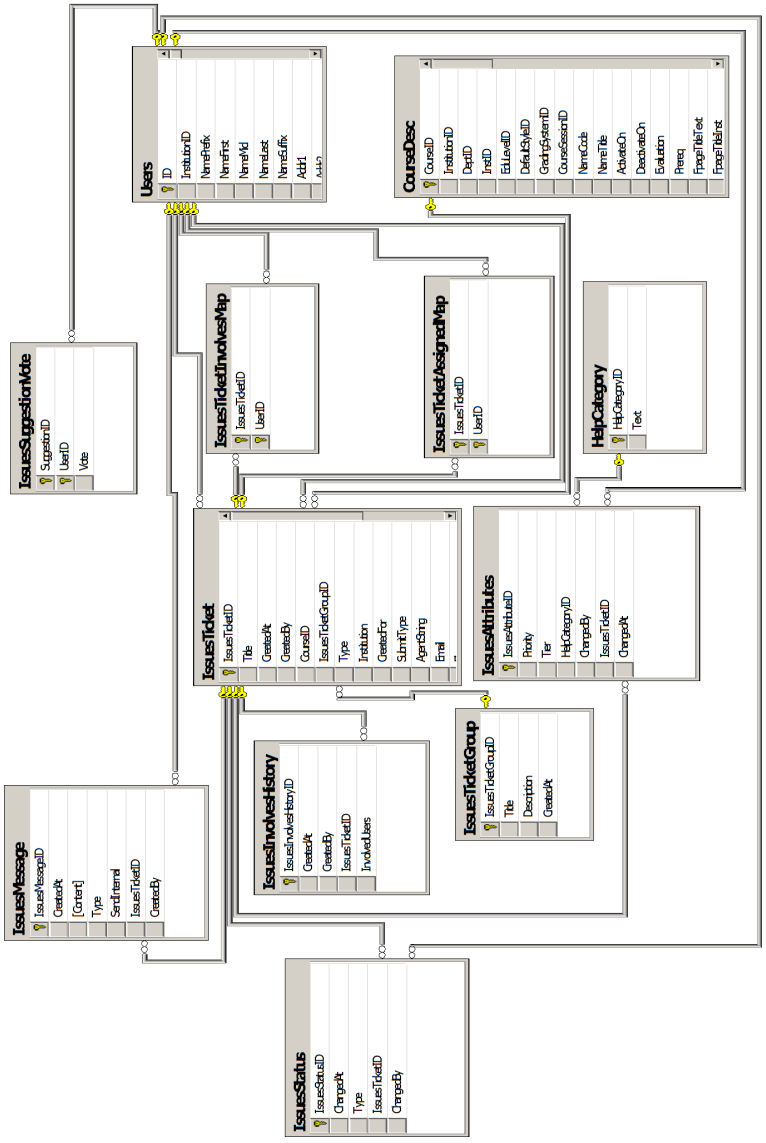
- AutoMapper
  https://github.com/AutoMapper
- Podio.NET client
  https://github.com/podio/podio-dotnet

A.2 Klientská časť aplikácie

- AngularJS
  https://angularjs.org/
- Bootstrap
  https://angular-ui.github.io/bootstrap/
- AngularUI Bootstrap
  https://angular-ui.github.io/bootstrap/
- AngularUI Router
  https://github.com/angular-ui/ui-router
- Restangular
  https://github.com/mgonto/restangular
- angular-bootstrap-switch
  https://github.com/frapontillo/angular-bootstrap-switch
- angular-no-captcha
  https://github.com/CodeDistillery/angular-no-captcha
- Angular Chart
  http://jtblin.github.io/angular-chart.js/
- lodash
  https://lodash.com/

Príloha B

Diagram relačného modelu databázy



Príloha C

Obsah priloženého CD

- súbor s popisom integrácie
- zdrojové súbory implementovaného systému
- zdrojové súbory technickej správy